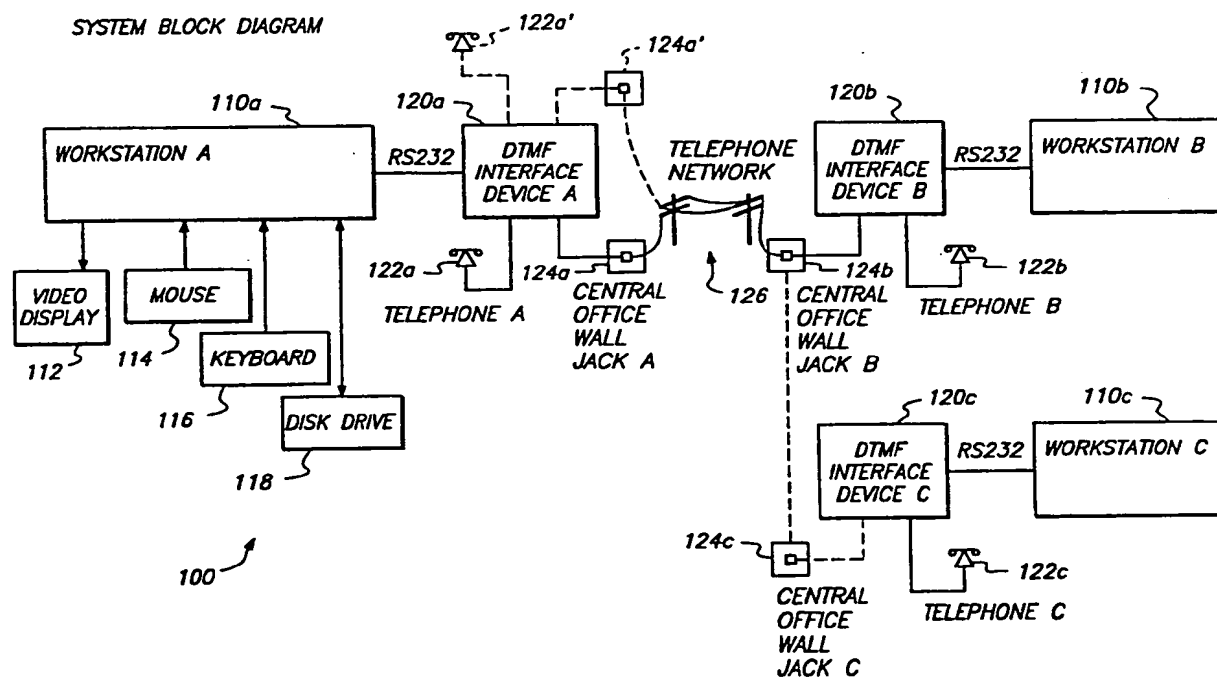




INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁵ : H04M 11/06	A2	(11) International Publication Number: WO 94/11983 (43) International Publication Date: 26 May 1994 (26.05.94)
(21) International Application Number: PCT/US93/11086 (22) International Filing Date: 16 November 1993 (16.11.93) (30) Priority data: 07/977,261 16 November 1992 (16.11.92) US (71) Applicant: OCTUS, INC. [US/US]; 9940 Barnes Canyon Road, San Diego, CA 92121 (US). (72) Inventors: BUSHNELL, Nolan ; 3860 Woodside Road, Woodside, CA 94062 (US). ANADY, Ed ; 13862 Otis Place, Poway, CA 92064 (US). LO, Roger ; 4018 Nobel Drive #304, San Diego, CA 92122 (US). HAY, Kevin ; 11146 Caminito Inocenta, San Diego, CA 92126 (US).		(74) Agents: SIMPSON, Andrew, H. et al.; Knobbe, Martens, Olson & Bear, 620 Newport Center Drive, Suite 1600, Newport Beach, CA 92660 (US). (81) Designated States: AT, AU, BB, BG, BR, BY, CA, CH, CZ, DE, DK, ES, FI, GB, HU, JP, KP, KR, KZ, LK, LU, LV, MG, MN, MW, NL, NO, NZ, PL, PT, RO, RU, SD, SE, SK, UA, UZ, VN, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG). Published Without international search report and to be republished upon receipt of that report.

(54) Title: DATA BURST SYSTEM FOR A TELEPHONE NETWORK



(57) Abstract

A system and method for transmitting data bursts across a telephone network. The present invention allows voice and data transfer during a single telephone connection. Voice communication during data transfer does not corrupt or terminate the transfer. The present invention includes an interface device for encoding and decoding DTMF tones into data. Caller information can be selectively transferred by a telephone user to the called person.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	GB	United Kingdom	MR	Mauritania
AU	Australia	GE	Georgia	MW	Malawi
BB	Barbados	CN	Guinea	NE	Niger
BE	Belgium	GR	Greece	NL	Netherlands
BF	Burkina Faso	HU	Hungary	NO	Norway
BG	Bulgaria	IE	Ireland	NZ	New Zealand
BJ	Benin	IT	Italy	PL	Poland
BR	Brazil	JP	Japan	PT	Portugal
BY	Belarus	KE	Kenya	RO	Romania
CA	Canada	KG	Kyrgyzstan	RU	Russian Federation
CF	Central African Republic	KP	Democratic People's Republic of Korea	SD	Sudan
CG	Congo	KR	Republic of Korea	SE	Sweden
CH	Switzerland	KZ	Kazakhstan	SI	Slovenia
CI	Côte d'Ivoire	LI	Liechtenstein	SK	Slovakia
CM	Cameroon	LK	Sri Lanka	SN	Senegal
CN	China	LU	Luxembourg	TD	Chad
CS	Czechoslovakia	LV	Latvia	TG	Togo
CZ	Czech Republic	MC	Monaco	TJ	Tajikistan
DE	Germany	MD	Republic of Moldova	TT	Trinidad and Tobago
DK	Denmark	MG	Madagascar	UA	Ukraine
ES	Spain	ML	Mali	US	United States of America
FI	Finland	MN	Mongolia	UZ	Uzbekistan
FR	France			VN	Viet Nam
GA	Gabon				

DATA BURST SYSTEM FOR A TELEPHONE NETWORKMicrofiche Appendix

A microfiche appendix containing computer source code is attached. The microfiche appendix comprises two (2) sheets of microfiche having 113 frames, including one title frame.

5 The microfiche appendix contains material which is subject to copyright protection. The copyright owner has no objection to the reproduction of such material, as it appears in the files of the Patent and Trademark Office, but otherwise reserves all copyright rights whatsoever.

Background of the InventionField of the Invention

15 The present invention generally relates to data transmission and, more specifically, to a system for transmission of data between two computers over a telephone line, using a protocol based on standard DTMF tones.

Background of the Technology

20 It is often desired to send data from a user's computer or workstation to another computer or to receive data from another computer at the user's computer. This communication is ideally desired in real-time, over a wide area, and with a diverse set of computer platforms. This type of data exchange is usually performed by use of standard modems over the telephone network.

25 However, real-time communications with modems is complex and requires both sides to have modem equipment and the proper settings. The user must know both the transmit and receive modem settings before transmitting. With a modem, a user cannot use a single telephone line for voice and data
30 communications simultaneously. The user, for example, would terminate a voice communication before being able to transmit data. Modems also do not associate contextual information with the transferred file. The ability to send and receive files is established but there is no ability to understand and
35 interpret the information sent.

In many cases, the amount of data to be communicated is small, which can be considered a data burst or set. In communication, a modem has an overhead or burden of a long setup time relative to a data burst to be transmitted. Since
5 the modem setup time is on the order of 5 to 15 seconds, and the data burst may be of a similar or shorter duration, a method of data communication eliminating the modem setup time is desirable. The need of a modem for data communication on a telephone line can be eliminated by using an existing common
10 standard, Dual Tone Multi Frequency (DTMF) codes or tones, for transmission. For short data sets, eliminating the modem and using DTMF codes for transmission leads to a faster effective data transfer rate.

During a telephone conversation, one person may ask the
15 other party for their address, title, facsimile (fax) number, or other information normally contained on a business card. Business card information can be considered a small data set suitable for DTMF tone transfer from one computer to another. It is desired to be able to transfer this information to the
20 requestor as quickly and accurately as possible, preferably to an easily indexed location such as a computer database. Limitations of prior methods include the following: mailing the information takes too long; facsimile only works if both parties have fax machines, but the information still needs to
25 be entered into the computer; and speech over the telephone may lead to misspelling or other mistakes. In addition, it would be advantageous to transfer the data during the middle of a voice conversation, for example, after the first party desires to place an order from the second party. The second
30 party sends a data burst containing his address or possibly some pricing information. The first party may send a data burst in response ordering a particular model number. Then, the voice conversation could resume until the call is ended with closing remarks.

35 Communicating short messages from computer to computer among coworkers or others in a Centrex or private branch exchange (PBX) environment is important in the business world.

Group scheduling and meeting reminders are just two instances of electronic-mail (E-mail) type messages. However, to use standard E-mail software, the user must have cables installed for a local area network (LAN) or a wide area network (WAN), along with the costly network operating software (OS). A way to communicate short messages for those businesses or organizations not having a regular LAN or WAN network is greatly desired.

A way of identifying a telephone caller before picking up the receiver, or otherwise answering, is important for many people. They may want to find a file or other information about the caller, or they may not want to answer the call. One potential means that is being introduced by the telephone companies is the use of Caller-ID. However, Caller-ID only provides the telephone number of the caller and no additional information. Many people could all be using the same telephone number, e.g., a company number, so automatic determination of the caller's name is not possible. Other systems provide a database of previous callers that is indexed after the call is answered and the name of the caller ascertained. The user would type in the name or part of the name to retrieve other information about the caller that has been prestored in the database. This type of system does not allow the user to avoid answering the telephone call depending on who is calling. Also, only prestored information is available. A means for providing a plurality of identification about the caller, including a call history, before answering the call is greatly desired.

In an office environment, a receptionist may announce a caller and ask the destination party or transferee whether the call should be transferred. An improved system would allow a call to be transferred in-context wherein the transferee will know who is calling and other information about the caller before the call is accepted and take appropriate action based on this knowledge. The call can then be refused, transferred to another person or back to the receptionist to take a message, or answered. Such a system that has little or no

human intervention would have great utility in the business world.

5 A telephone assistance apparatus for the hearing impaired, including a tone decoder responsive to DTMF output of a telephone receiver, is disclosed by Fowler, et al., (U.S. Patent No. 4,608,457). The decoder converts the DTMF output into a digital binary code corresponding to the key combinations depressed by a sender on the sender's telephone equipped with a DTMF generator (Touch Tone type telephone).
10 A letter, numeral, punctuation mark, or one of several possible words are sent by two keystrokes on the sender's telephone keypad.

Such a system includes a number of significant limitations. The sender must either have a table or chart defining the keystroke combinations, or must be told by the hearing impaired person of the 58 recognized keystroke combinations. The process of sending a message or data is slow because the sender manually generates the keystroke combinations. Also, the DTMF encoded data transfer is
20 unidirectional; the hearing impaired person communicates by voice only.

A data transmission system for interfacing with a telephone line is disclosed by Brule, et al., (U.S. Patent No. 4,937,853). The system is used for entering bar code data
25 into a remote computer by standard DTMF telephone tones. An interactive telephone calling network interfaced with the remote computer sends a voice menu to the user. The user enters an identification number and passcode by either using bar code data, the system keypad, or the keypad of an optional telephone. Thereafter, data is entered by use of a bar code
30 reader. This data is converted to ASCII characters by the system, which are then subsequently converted to DTMF tones for transmission. The system has a tone receiver for receiving information from the remote computer for display on a LCD display to communicate with the user when the telephone
35 is not provided. The system does not allow both voice and data communication by the user in one connection. The data

must be converted to bar code format sometime before the telephone call is initiated or the data must be keyed in one character at a time on the user's keypad.

5 To facilitate the creation of a data message before initiating a telephone call, a prior device uses a keyboard to enter the message into the device. A portable terminal used for brief queries to a database on a large central computer using a telephone for communication over the telephone network is disclosed by Dayton, et al., (U.S. Patent No. 4,799,254).
10 Message characters are typed on the keyboard, digitally coded and stored in memory. After a dial-up connection to the computer, the stored message is converted to a code using unique pairs of standard DTMF tones provided by a tone generator to a speaker connected to a telephone mouthpiece.
15 The computer responds to the user either by recorded or synthetic speech to a telephone earpiece or by DTMF tones to an optional microphone connected to the earpiece. The DTMF tones are converted in the terminal to digital codes for storage in memory and later display on a eight-character LCD
20 display. The telephone at the user end is not used for outgoing voice since the telephone line at the destination is connected to the central computer. Therefore, both voice and data communication in a single connection cannot be accomplished.

25 Thus while the preceding patents discuss converting barcode data, telephone keypad keystrokes, or stored ASCII data to DTMF codes for transmission on a telephone network to a remote central computer or a telecommunications device for the hearing impaired, none of the patents addresses having
30 both voice communication and data bursts together during a single telephone connection. Prior patents do not address having a tone encoder/decoder combination to carry out a conversation and data exchange between two people.

35 Consequently there is a need for a system that permits wide area communication using telephone lines and DTMF data encoding for a data burst in combination with voice over a single telephone connection. Such a system should accommodate

users on a wide variety of computer platforms, as well as permitting data bursts in either direction between two users or both directions in a serial fashion. Such a system should also be inexpensive and easy to install and use.

5

Summary of the Invention

The above-mentioned needs are satisfied by the present invention which includes a system and method for sending data bursts or messages across a telephone line between computer users.

10

Brief Description of the Drawings

Figure 1 is a system level block diagram of a presently preferred embodiment of the present invention showing a sending workstation, a first destination workstation, and a second optional destination workstation;

15

Figure 2 is a functional diagram showing the hierarchy of the software for a send process or operation that is executed in a workstation of Figure 1;

Figure 3a is a block diagram showing the main components of the DTMF interface device of Figure 1;

20

Figure 3b is a schematic diagram of the DTMF interface device of Figure 1;

Figure 4 is a flowchart of a user interface layer or shell that is shown in Figure 2;

Figure 5 is a flowchart of the STARTDATA function of Figure 4;

25

Figure 6 is a flowchart of the SENDDATA function of Figure 4;

Figure 7 is a flowchart of the ENDDATA function of Figure 4;

30

Figure 8 is a flowchart of the BEGINBOXDATA function of Figure 5;

Figure 9 is a flowchart of the BOXDATA function of Figure 6;

Figure 10 is a flowchart of the ENDBOXDATA function of Figure 7;

35

Figure 11 is a flowchart of a TOOLS LAYER-RECEIVE process

that runs on the system of Figure 1 as shown in Figure 2;

Figure 12 is a flowchart of the TELCALLBACK function of Figure 11;

5 Figures 13a and 13b are a flowchart of a Business Card Transfer function shown in Figure 2, and which is stored and executed in the data burst system shown in Figure 1;

Figure 14 is an example of a screen display presented to a user when the telephone module is first activated as a part of the function shown in Figure 13;

10 Figure 15 is an example of a screen display illustrating information for an outgoing call on line 1 as presented by the function shown in Figure 13;

Figure 16 is an example of a screen display presented to a user when the Phone Menu is opened as performed by the function shown in Figure 13 ;

15 Figure 17 is an example of a screen display illustrating information selected for a Business Card Transfer as performed by the function shown in Figure 13;

Figure 18 is an example of a screen display presented to a user when the received burst already exists in a database as presented by the function shown in Figure 13;

Figure 19 is an example of a screen display presented to a user when the received burst does not exist in the database as presented by the function shown in Figure 13; and

25 Figure 20 is an example of a screen display presented to a user after a toggle to View Original has been done presented by the function shown in Figure 13.

Detailed Description of the Preferred Embodiment

30 The following detailed description of the preferred embodiments presents a description of certain specific embodiments to assist in understanding the claims. However, the present invention can be embodied in a multitude of different ways as defined and covered by the claims.

35 Reference is now made to the drawings wherein like numerals refer to like parts throughout.

I. System Overview

The data burst system of the present invention includes means to communicate short bursts of information over the most ubiquitous network in the world, the telephone network. Current network technology often requires specific hardware for each workstation, special wiring between stations and complex protocols to transmit data. The present system uses the telephone network to transmit data encoded into dual tone multi frequency (DTMF) signals and then decode the signals into standard ASCII characters for interpretation by application software. The means by which the data is interpreted and the applications that use this data are described hereinbelow.

Each workstation is equipped with a DTMF Interface Device, also referred to herein as the network device, that integrates the telephone with the workstation. The network device can control such telephone operations as dialing, taking the telephone line off hook and reading the DTMF codes entered by the user. In addition, the network device monitors telephone status, e.g., off hook, ringing, connected, and so forth, and also communicates with the workstation via a RS232 serial connection, a well-known standard in the computer and communication technologies, and data burst protocol to be described hereinbelow. The network device can determine the noise on a telephone line and reduce the time in between DTMF codes to speed up transmission, and it can control the volume of the lines. Also, the network device has the ability to convert DTMF codes to ASCII and vice versa.

Figure 1 is a block diagram illustrating an exemplary data burst system 100. The system 100 includes at least two computers or workstations 110a, 110b, and shown with a third optional workstation 110c which, for instance, may be IBM PC compatible computers, each having a processor that is at a minimum an Intel 80386 class microprocessor, and a memory (not shown). Additionally, each of the workstations 110 includes a plurality of input/output (I/O) devices. For example, each of the workstations 110, as specifically shown at the

workstation 110a, connects to a video display 112, a pointer device such as a track ball or mouse 114, a keyboard 116 and a secondary storage device such as a disk drive 118. It should be noted that, although the following description is provided as if a message sender, or user (not shown), operates the workstation 110a via the input/output devices 112-118, users also employ the workstations 110b and 110c which are also understood to communicate with similar input/output devices. Of course, any number of workstations 110 may be provided in the system 100.

A DTMF Interface Device 120a (also referred to as box or interface box) is connected to workstation 110a at a communications port (not shown) via a RS232 serial interface. The device 120a also connects to a telephone 122a and a signal line connecting to a telephone jack 124a. Workstation 110b connects via a RS232 serial interface to a DTMF Interface device 120b. Device 120b also connects to a telephone 122b and a signal line connecting to a telephone jack 124b. Similarly, workstation 110c connects via a RS232 serial interface to a DTMF Interface device 120c. Device 120c also connects to a telephone 122c and a signal line connecting to a telephone central office wall jack 124c.

The system 100 thus comprises a plurality of data burst subsystems wherein each subsystem can transmit/receive voice and data signals. Each subsystem may also be referred to herein as a "databurst system".

Some workstations may be at remote locations from each other as illustrated in Figure 1. The workstations at remote locations are connected via telephone lines, which are typically not directly wired, but instead are switched via a switching network such as a public telephone network 126.

Figure 2 illustrates a software hierarchy for the presently preferred workstation on 110. The software hierarchy carries out the data burst send and receive processes. A set of integrated application software running under a unified application system shell environment, referred to hereinafter as a shell 142, both initiates and interprets

the transmitted data. The shell 142 preferably operates in conjunction with a windows environment 140 such as Windows 3.1 licensed by Microsoft. The integrated application software includes modules, e.g., telephone module 144, with associated submodules or functions, e.g., Business Card Transfer 146. The shell 142 allows for easy upgrades of the data burst system by installing drop-in modules or applications. Once installed, the shell 142 automatically adds that module to a list of icons that appear on the left side of every shell window presented on the video display 112 (Figure 1). In the presently preferred system 100, database (not shown) is to the shell module 142 using an object oriented accessible filing system. In the presently preferred embodiment, the database utilized is db_VISTA available from Raima Corporation, which is a relational database that supports variable length records.

The workstation module software 144 is divided into three layers to facilitate ease of use of the telephone 122 and isolate the applications from the host system or workstation 110. These layers include a tools layer 150, a manager layer 148, and an application layer 146, which is also called a user interface (UI). The first two layers are frequently referred to as tools and manager, respectively. The tools layer 150 communicates directly with the device 120 (Figure 1) by use of a serial connection. Using serial connections provides for easy migration to a variety of platforms, e.g., Macintosh®, Sun®, and IBM PC. The responsibility of the tools layer 150 is to communicate with the hardware directly using the commands and protocols of the hardware. The manager layer 148 interprets the communication with the device 120 and informs the module 144 of changes in status. The separation of a manager layer from the tools layer permits the swapping out of the tools module and hardware, while maintaining a consistent, common application program interface (API) for the User Interface regardless of hardware type. The user interface layer 146 provides a graphical interface for the user to easily accomplish the data transfers.

For example, consider workstation 110a as the origin workstation, device 120a as the origin device, workstation 110b as the destination workstation, and device 120b as the destination device. (Of course, a message transfer in the reverse direction is equally possible.) The user interface 146 initiates data transmission, while the manager layer 148 breaks the data into manageable packets and downloads the data to the device 120a (Figure 1) via the tools layer 150. The device 120a places the data in an envelope or packet, describes the type of transfer and its checksum, and then converts each ASCII byte to two DTMF codes. The data is then transmitted to the destination device 120b via the telephone lines and decoded to ASCII data. The destination device 120b uploads the data to the destination host workstation 110b, where its manager recombines the packets and passes the information to the module 144. At every transmission or conversion, a checksum is provided to verify data integrity. If an error is detected, a not acknowledge (NACK) is passed back to the originator along with information on where the error occurred. If the data is successfully sent, a single acknowledge (ACK) is transmitted back to the originator.

Figure 3a is a block diagram of the major circuits and their interconnection for the DTMF Interface device 120 (Figure 1). Figure 1 illustrates device 120 with one telephone 122 and one telephone line connected to the wall jack 124, device 120 but includes connections for a telephone line 1 380, a telephone 1 122, a telephone 2 384, and a telephone line 2 386. A user of the system 100 may optionally choose to use either one or two telephones, e.g., 122a and 122a' and one or two lines per device 120. For example, the user could have one telephone and two lines. The device 120 can even be operated without a telephone, but the user is then limited to speakerphone rather than handset conversations. Line 1 380 interconnects a wall jack 124a to a telephone line interface 1 402, while line 2 386 interconnects another wall jack 124a' to a telephone line interface 2 404.

Telephone 1 122 connects through a switch 405, controlled

by the workstation 110 connected to the device 120, to either a telephone interface 1 406 (normal mode) or to a power supply 414 (local mode). Telephone 2 122' connects through a switch 407, also controlled by the workstation connected to the device 120, to either a telephone interface 2 408 (normal mode) or to the power supply 414 (local mode). As blocks 402, 406 and 410 are substantially the same as blocks 404, 408 and 412, only the circuitry with respect to telephone 1 122 will now be described.

The telephone line interface 402 utilizes a "loop start" type interface circuit that is well known in the telephone technology. This circuit generates at least 20 ma of loop current needed by the telephone company to detect an off-hook condition of the telephone handset and respond with a dial tone. The telephone line interface 402 provides adequate filtering and protection from lightning and voltage spikes. The telephone line interface 402 sends and receives audio signals, which include Touch Tones or DTMF tones™, on an audio path bidirectionally with the telephone interface 406 and with a tone circuit 1 410. The telephone line interface 402 also generates a ring signal on a ring signal line to a microprocessor complex 420.

The telephone interface 406 has two modes of operation as follows:

(1) Normal mode which is default even in the event of power loss to the network device 120. Normal mode just passes the telephone signals through to the bidirectional audio path with the tone circuit 410.

(2) Local mode in which the network device 120 is powering the external telephone 122 through the power supply 414. The external telephone 122 is treated just like a microphone and speaker to the audio circuits, and thus can be routed in or out of the device 120 as desired using an audio path to an audio switch 418. In local mode, the phone 122 will not ring. This is useful in the case where the device 120 first analyzes Caller Identification (ID) information, if present, after the first ring of a telephone call and notifies

the phone module 144 (Figure 2) before the user hears the phone ring.

5 In either mode, the microprocessor complex 420 monitors the state of the telephone handset for on or off hook condition via handset signal lines from the telephone interface 406. This information is passed on to the telephone module 144 in the workstation 110, which takes further action.

10 The device 120 has two independent tone circuits channels 410 and 412. Each tone circuit is physically tied to its respective telephone line 380, 386 and wall jack 124, 124'. The tone circuit 410 has three main functions:

- (1) DTMF (Dual Tone Multi-Frequency) Transmission
- (2) DTMF Receiver
- (3) Call Process Filter

15 The transmission section is capable of generating all 16 standard DTMF tone pairs with low distortion and high accuracy. All frequencies are derived from a 3.58 Mhz crystal. The sinusoidal waveforms for the individual tones are digitally synthesized using programmable dividers and
20 switched capacitor digital-to-analog converters.

The well known DTMF tones include two sinusoidal signals simultaneously transmitted. One signal is selected from a group of four high frequencies, and the other signal from a group of four low frequencies. The duration of the two
25 frequency signal (burst) is 40 milliseconds (mS) while an intersignal or pause duration is 30 mS +/- 10 mS. The standard signal plus pause interval for Autodialers, Central Office applications and also the device 120 default is 51 mS +/- 1mS. The pause duration is software adjustable from 20 mS to 200 mS
30 with a 'S' register command (ATS11=n). On a quiet telephone line, both dialing and data bursts show a speed improvement by adjusting this register value.

The telephone keypad numbers 0 to 9, symbols "*" and "#", and the letters "A", "B", "C", "D" are represented by pairs of
35 simultaneously transmitted signals of the frequencies (in Hertz) indicated in Table 1. These values are the standard DTMF tones used by the telephone companies.

TABLE 1

		<u>1209</u>	<u>1336</u>	<u>1477</u>	<u>1633</u>
	697	1	2	3	A
	770	4	5	6	B
5	852	7	8	9	C
	941	*	0	#	D

The DTMF receiver section of the tone circuit 410 includes a tone chip that converts incoming DTMF tones into a digital format. The digital data is sent or received on a multi-signal digital path connected to the microprocessor complex 420. DTMF data (0-9, *, #, A, B, C) is converted to 8-bit digital data and vice versa, as indicated in Table 2.

TABLE 2

DTMF to 8 Bit Data

20	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Second D	1	2	3	4	5	6	7	8	9	0	*	#	A	B	C	Digits
First																
25 D	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
1	10	11	12	13	14	15	16	17	18	19	1a	1b	1c	1d	1e	1f
2	20	21	22	23	24	25	26	27	28	29	2a	2b	2c	2d	2e	2f
3	30	31	32	33	34	35	36	37	38	39	3a	3b	3c	3d	3e	3f
4	40	41	42	43	44	45	46	47	48	49	4a	4b	4c	4d	4e	4f
30 5	50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d	5e	5f
6	60	61	62	63	64	65	66	67	68	69	6a	6b	6c	6d	6e	6f
7	70	71	72	73	74	75	76	77	78	79	7a	7b	7c	7d	7e	7f
8	80	81	82	83	84	85	86	87	88	89	8a	8b	8c	8d	8e	8f
9	90	91	92	93	94	95	96	97	98	99	9a	9b	9c	9d	9e	9f
35 0	a0	a1	a2	a3	a4	a5	a6	a7	a8	a9	aa	ab	ac	ad	ae	af
*	b0	b1	b2	b3	b4	b5	b6	b7	b8	b9	ba	bb	bc	bd	be	bf
#	c0	c1	c2	c3	c4	c5	c6	c7	c8	c9	ca	cb	cc	cd	ce	cf
A	d0	d1	d2	d3	d4	d5	d6	d7	d8	d9	da	db	dc	dd	de	df
B	e0	e1	e2	e3	e4	e5	e6	e7	e8	e9	ea	eb	ec	ed	ee	ef
40 C	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	fa	fb	fc	fd	fe	ff

7 of the 8 Bits (128 of 256) are used for standard ASCII, e.g., Digits "4" "1" = ASCII "A",
 45 Digits "7" "c" = Hexadecimal 7f = ASCII "DEL"

An extended Telephone Keypad is utilized for complete 1 of 16 DTMF codes.

5 The call process filter section of the tone circuit 410 helps

the microprocessor complex 420 discriminate and count useful tones

10 of interest:

Dial Tone	350 Hz	continuous
Audible Ring	440 Hz	2 Sec. on / 4 Sec. off
Busy	440 Hz	0.5 Sec. burst

15

The power supply 414 is an UL/CSA approved 11-13 VAC, 1 amp supply which powers the internal circuits. The following signals are internally generated:

20 -35 to -40 VDC for powering telephone handsets
12 VDC for speakerphone and some analog circuits
5 VDC for microprocessor circuits
+/-8 VDC for serial communications
-2 VDC for audio amplifiers

25 The audio switch 418 includes bidirectional audio path connections to a speakerphone 424, an audio in/out block 428, the telephone handset 122 through the telephone interface 406, the telephone line 380 through the telephone interface 406 and telephone line interface 402, and a one-way
30 connection to a caller identification (ID) circuit 430. The audio switch 418 also includes a digital bidirectional path to the microprocessor complex 420.

35 An 8 x 8 crossbar switch (not shown) is utilized by the audio switch 418. The phone lines 380, 386 are treated as having separate in and out lines. The telephone handsets 122, 122' are treated as having separate in and out lines. The speakerphone 424 has a separate in and out line. The audio block 428 has both 2-channel in and out lines. The Caller ID circuit 430 has only an in line. All the above
40 combinations, other than Caller ID, are negotiated by the telephone module 144 (Figure 2) as the main routing mechanism. Any input to any output is possible. The 8 x 8

crossbar switch is illustrated by the matrix of Table 3. The letters A, B, C, and D represent features of the present embodiment of the invention which are, respectively, music on hold, silence during hold, device conference connect lines, and speakerphone outgoing call. Of course, other features may be possible.

TABLE 3

	<u>0</u>	<u>1</u>	<u>2</u>	<u>3</u>	<u>4</u>	<u>5</u>	<u>6</u>	<u>7</u>
10 Tel Line Out1	D				C			
Tel Line Out2	D			C				
Ground				B	B			
Audio In 1				A	A			
Audio In 2				A	A			
15 Handset Out 1								
Handset Out 2								
Spkrphone Out				D	D			

where columns are defined as follows:

- 0 = Speakerphone In
- 1 = Audio Out 1
- 2 = Audio Out 2
- 3 = Tel Line In 1
- 4 = Tel Line In 2
- 5 = Handset In 1
- 6 = Handset In 2
- 7 = Caller ID

The microprocessor complex 420 utilizes a traditional 65C02 8-bit microprocessor design with 8K RAM, 8K Flash ROM, a versatile interface adapter (VIA) 6522 chip including two 16-bit timers and 20 lines of input/output (I/O), and an address decode 74HC138 chip running at 1.84 Mhz bus speed. The control flow of the complex 420 includes polling and interrupt mechanisms. The microprocessor complex 420 also includes a communications port 422 having two asynchronous communication interface adapter (ACIA) 6551 chips and a RS232 interface MAX232 chip available from Maxim Integrated Products, that connects to the workstation 110 (Figure 1) and optionally to another device 120. The device 120 to device 120 connection allows the AT+LINK command to be used for

device to device communication. In another embodiment, the components of the microprocessor complex 420 may be combined together in one semi-custom package.

5 The operation of the 65C02 microprocessor is controlled by a program written in assembly language. Some of this code is listed in the attached Microfiche Appendix with sections titled "const.src", "kernal.src", and "MAIN.src". One skilled in the technology will recognize that the operation of the device 120 can be controlled by a number of different
10 processors, languages and circuitries.

Caller ID or Caller Number Delivery (CND) is a feature in a General Switched Telephone Network (GSTN). This service is provided by the telephone company and allows called
15 customers to receive a calling party's number during the ring cycle. The device 120 has one receiving circuit that it shares between the two lines. Characteristics of Caller ID include:

	<u>Characteristic</u>	<u>Description</u>
	Link Type	Simplex, two wire
20	Transmission scheme	Analog, phase-coherent FSK
	Logical 1 (mark)	1200 +/-12 Hz
	Logical 0 (space)	2200 +/-22 Hz
	Transmission Rate	1200 Bps

25 A sequence of steps for Caller ID (CID) follow using line 2 as an example. The workstation 120 causes the switch 407 to be at the position shown in Figure 3a, which sets the telephone handset 122' in Local mode. Local mode allows
30 Caller ID operation. The device 120 senses an incoming ring on line 2 and switches the CID circuit 430 to receive signals from line 2. The device 120 reads the CID data and then resets the CID switch setting back to Normal mode. The device 120 sends the CID information to the telephone module
35 144 (Figure 2) for further action. The telephone module 144 then sends a command to switch the telephone handset 122' to Normal mode. The audio in/out block 428 has two-channel audio inputs and two-channel audio outputs to pass audio signals to and from the phone lines. In a simple

implementation of the block 428, an external music source could be used for music on hold. Also, computer generated sound bites can be placed on the phone lines via block 428. Recording phone conversations is also made possible by line
5 level ready analog signals at block 428.

In another embodiment, a SOUND BLASTER™ by Creative Labs, Inc., sound card product is utilized with the device 120 to implement a digital answering machine.

In yet another embodiment, with the addition of a
10 daughter board 432, the device 120 becomes Fax/Data/Voice ready without the need of the sound card unless simultaneous Fax and Digital voice recording is required. The daughter
15 board 432 is a small printed circuit board (PCB) which plugs into the DTMF Interface device 120. The daughter board is a co-processor design which memory maps into the device 120.

Unlike Centrex conference, two line conference establishes connection with a minimum of two external parties and essentially hooks them together with the Audio Switch 418. The main advantage over 2-line telephones with
20 conference features is the ability to adjust incoming and outgoing volume levels on both lines.

The DTMF Interface device 120, as directed by the telephone module 144, is always monitoring Touch Tone information. In its simplest form, device 120 could be used
25 for an order entry or voice prompting system. The next level of complexity assumes two shell users are active in this mode with a robust protocol, error correction and high speed tones or, in the present invention, short pauses between tones provides a Data Exchange Vehicle. While this Data Exchange
30 Vehicle could be compared to a low speed modem, the device 120 provides more capability.

Internal to the tone circuit 410 is a decoder which employs digital counting techniques to determine the frequencies of the incoming tones, and verify that they
35 correspond to a standard DTMF frequency. An averaging algorithm protects against tone simulation by extraneous signals (such as voice), while tolerating small deviations in

frequency. The algorithm provides an optimum combination of immunity to talkoff with tolerance to interfering frequencies (third tones) and noise.

5 An example of data flow through the device 120 using workstation 110a (Figure 1) as the origin and workstation 110b as the destination follows. The RS232 links between the workstations 110 and devices 120 carry digital data at 9600 bps. The telephone network 126 is a voice grade telephone line - analog link. The workstation 110a sends a packet of
10 ASCII data with a variable length to a maximum of 512 bytes in the presently preferred embodiment. This includes length of data, data, checksum and terminator.

The DTMF Interface device 120a receives this ASCII data and places it into its RAM buffer in the microprocessor
15 complex 420 and starts the conversion process to an analog signal in the tone circuit 410 or 412. This analog signal is a digitally synthesized sinusoidal waveform converted using switched capacitor digital-to-analog converters, and is conforming to one of 16 standard DTMF tone pairs. The device
20 120a begins transmitting this analog data to device 120b. Due to the possibility of telephone line noise, length, and echo, the transmission speed typically varies from 100 to 180 bits per second. Both amplitude and duration are variables in the analog data and are software correctable.

25 The DTMF Interface device 120b then receives the incoming analog data. The high and low tone-pairs are separated by applying the DTMF signal to the inputs of two sixth-order switched bandpass filters in the tone circuit 410 or 412. The tone circuit decoder then determines the
30 frequency by employing digital counting techniques which yield data that is further processed and packetized to 9600 bps ASCII data suitable for passing on to the workstation 110b. This process can go in either direction.

35 Figure 3b is a schematic diagram of the DTMF interface device 120 shown in Figure 1 and 3a. Table 4 is a list of components corresponding with the schematic diagram of Figure 3b.

TABLE 4

5

10

15

20

25

PART NO.	DEVICE TYPE	DESCRIPTION
C69	CAP	ELECT, AXIAL, 330uf, 6.3v, 20%
C64	CAP	ELECT, AXIAL, 1000uf, 25V, 20%
C22, 38	CAP	ELECT, AXIAL, 220uf, 100v; 20%
C66, 70, 71	CAP	ELECT, RAD, 330uf, 50V, 20%
C67	CAP	ELECT, RAD, .47uf, 50v, 20%
C23, 37	CAP	ELECT, RAD, 4.7uf, 50v, 20%
C1-7, 17-19, 21, 24, 30-32, 35, 36, 39-45, 49, 52, 56, 65, 68, 73-90, 92-97	CAP	FIXED AX CER 50V +80-20%, .1 MFD
C57	CAP	CER, RAD, .01uf, 50V, 5%
C62	CAP	CER, RAD, .068uf, 50V, 5%
C20, 34	CAP	CER, RAD, 33uf, 100v, 5%
C12, 13, 25, 26	CAP	CER, RAD, 270pf, 1000v, 10%
C50, 51, 54	CAP	FILM, RAD, .068uf, 50v, 5%
C15, 28	CAP	METAL, RAD, .47uf, 250v, 10%
C14, 27	CAP	METAL, RAD, .068uf, 250v, 10%
C48, 58, 59	CAP	TAN, RAD, 4.7uf, 6.3v, 10%

-21-

5

10

15

20

25

C16, 29, 33, 46, 47, 63, 91	CAP	TAN, RAD, 1uf, 16v, 10%
C8, 9, 10, 11	CAP	TAN, RAD, 10uf, 16v, 10%
C53, 55, 60, 61, 72	CAP	TAN, RAD, 47uf, 16v, 10%
R9, 42	RES	CARBON, 0 ohms, 1/4W, 5%
R27, 32, 33, 60, 65, 66, 102, 107	RES	CARBON, 100 Ohms, 1/4W, 5%
R15, 19, 20, 23, 34, 37, 48, 52, 53, 56, 67, 70, 91, 98, 99, 103, 104, 108, 109	RES	CARBON, 10K ohms, 1/4W 5%
R1, 13, 21, 22, 29, 46, 54, 55, 62, 84, 100, 105	RES	CARBON, 100K, 1/4W, 5%
R35, 39, 68, 72	RES	CARBON, 12K, 1/4W, 5%
R92	RES	CARBON, 1.5K, 1/4W, 5%
R16, 49, 86	RES	CARB FILM; .25W 5%, 18K OHMS
R30, 31, 63, 64, 90, 110, 111	RES	CARBON, 200 Ohms, 1/4W, 5%
R12, 45, 88	RES	CARBON, 200K, 1/4W, 5%

	R2, 77, 78	RES	CARBON, 2M Ohm, 1/4W, 5%
	R38, 71, 89, 101, 106	RES	CARBON, 22K Ohm, 1/4W, 5%
	R95	RES	CARBON, 24K, 1/4W, 5%
5	R80	RES	CARBON, 30K, 1/4W, 5%
	R7, 10, 26, 40, 43, 59, 82	RES	1/4 w, 5%, 3.3k OHMS
	R24, 57	RES	CARBON, 360K, 1/4W, 5%
	R87	RES	CARB FILM; .25W 5%, 4.3K OHMS
10	R4-6, 11, 14, 18, 25, 44, 47, 51, 58, 79, 83, 85, 96, 97	RES	CARB FILM; .25W 5%, 4.7K OHMS
15	R73-76, 93, 94	RES	CARBON, 47K, 1/4W, 5%
	R28, 61	RES	CARBON, 620 Ohm, 1/4W, 5%
	R81	RES	CARBON, 91K, 1/4W, 5%
	R8, 41	RES	CARBON, 10 Ohm, 1/4W, 1%, METAL
20	D1, 2, 6, 9, 13, 29, 30, 32, 33	DIODE	AXIAL, 1N914
	D16-19, 24, 25, 28	DIODE	1N4001
	D20-22	DIODE	1N4002

D3, 4, 7, 8, 10, 11, 14, 15	DIODE	1N4728, ZENER, 5%
D5, 12	DIODE	1N4748, ZENER,
Q5, 6	TRANSIST.	NPN, 2N3904, TO-92
Q1-4	TRANSIST.	2N4403
U3	IC	74HC138, 3 to 8 LINE DECODER
U4	IC	74HC139
U1	IC	74HC240
U10	IC	MAX232, RS-232 DRIVER/RECEIVER
U2	IC	MICROPROCESSOR, 65C02, 2MHZ
U8, 9	IC	6551 ASYNC COM INTERFACE ADAPTER, 2MHZ
U7	IC	6522 PERIPHERAL INT. ADAPTER, 3MHZ
U5	IC	5565 SRAM, 8Kx8, 120nsec
U6	IC	CAT28C64, EEPROM, 200nsec
U13, 14, 19	IC	LM324, OPAMP
U25	IC	LM7805, 5v REGULATOR
U24	IC	LM7812, 12V REGULATOR
U15, 18	IC	M8880, TRANSCEIVER, DTMF
U21	IC	SC11120, CALLER ID
U22	IC	MC34018, SPEAKER PHONE
U20	IC	MT8809, INTERFASE ANALOG SWITCH 8x8

- 24 -

U11, 16	IC	EEPOT, 10K
U12, 17, 23	IC	EEPOT, 50K
	LED	GREEN
	LED	ASSY(4), GREEN
ISO1-4	OPTI- ISOLATOR	
FB1-8	INDUCTOR	WIRE BEAD, .138 DIA. X .175L
T1-4	TRANS- FORMER	A19N-HH-1A/1
SW1	SWITCH	SPST, PUSH
K1-4	RELAY	DPDT
RV1,2	MOV	250V
Y1	XTAL	1.8432MHz
Y2	XTAL	3.57954MHz
SPK1	SPEAKER	32 Ohm, 40mm
MIC1	MICRO- PHONE CARTRIDGE	
J1	CONN	DIN, CIRCULAR, 5PIN
J2,3	CONN	PHONE, RJ-45, 8-PIN
J4, 5	CONN	PHONE, RJ-11, 6-POS, 4-PIN
J6, 7	CONN	JACK, STEREO, 5-PIN
J8	CONN	JACK, DC POWER, 3-PIN

II. Data Burst Protocol

This section describes the software interface between the DTMF Interface device 120 (Figure 1) and the shell software. Wherever possible, the device 120 emulates a Hayes compatible modem.

A. Device to Workstation Commands

The general format of a command from the device 120 to the workstation 110 in the presently preferred embodiment is as follows:

|DLE | MessageID | Length | Data | Checksum|

Where:

<u>Field and length of Field</u>	<u>Description</u>
Data-Link Escape (DLE) (8 bits)	Hex value 10
MessageID (8 bits)	Specifies Message type and Phone Line number
Message type (5 Most Significant bits)	
Line number (3 Least Significant bits)	
Length (16 bits)	Specifies the length of the data plus checksum
Data (n bytes)	The body of data
Checksum (8 bits)	An 8 bit checksum

The checksum is calculated by taking the sum of the ASCII values of the data and then exclusive-ORing (XOR) the sum with the value 0xFF. The XOR boolean operator is well-known in the computer technology.

To preserve space, this format is violated when the message type is Status. In this case, the Length field is replaced with the Status word, and the Data and Checksum fields are not used.

1. Message Types

The following is the list of Message types and their values:

-26-

	<u>Message</u>	<u>Value</u>	<u>Line Number Info.</u>	<u>Description</u>
	Status	0x01	yes	Device Status
	Link	0x02	no	Link Upload Request
	Burst	0x03	yes	Burst Detected
5	DTMF Data	0x04	yes	DTMF Data in Buffer
	ANI	0x05	yes	Automatic Number ID
	Burst Err	0x06	yes	Burst Error
	Burst Recd	0x07	yes	Burst successfully transmitted
10	Link Err	0x08	no	Link Error
	Link Recd	0x09	no	Link successfully transmitted

2. Status Byte Definition

15 The bit assignments for the Status bytes and their definitions are:

FIRST BYTE

	Bit 0:	Ring	The phone is ringing
	Bit 1:	Off Hook	The handset is off hook
	Bit 2:	Busy	The Line is Busy
20	Bit 3:	Dial Tone	The Line has a Dial Tone
	Bit 4:	Dialing	The Line is Dialing out
	Bit 5:	DTMF In	There are Characters in the DTMF Buffer
	Bit 6:	Line Off Hook	The Line is off Hook
25	Bit 7:	Device Error	The Device has detected an error

SECOND BYTE

30	Bit 0:	Link Request	A Link has been requested
	Bit 1:	Link Active	Link is active
	Bit 2:	Burst Request	A Burst is Requested
	Bit 3:	Burst Active	Burst is Active
35	Bit 4:	Connection	Connection Detected After Dial

-27-

Bit 5: Time-out Either no connection
detected after ringing
stops or no dial tone
detected after off-hook

5 Bit 6: Unused

Bit 7: Unused

10 The Status message will be sent whenever a change in state
occurs for any status or following an ATZ command (described
below).

B. Workstation to Device Command Set

15 In the presently preferred embodiment the following standard
HAYES AT Commands are supported. All commands must end with
the Carriage Return (0x0D) character.

	<u>Command</u>	<u>Description</u>
	ATD	Dialing
	ATDT	Tone Dialing
20	ATDP	Pulse Dialing
	<i>Dialing Modifiers</i>	
	T	Tone
	P	Pulse
25	W	Wait for Dial Tone. If dial tone is not sensed within 15 seconds, the call is terminated and the Device returns to command mode.
	,	Pause for the number of seconds stored in Register 8. Default is 2 seconds.
30	!	Flash
	0-9#* A-D	Tones
	ATH0	On Hook
	ATH1	Off Hook
	ATIO	Product ID
35	ATI1	ROM Checksum

	ATI3	Software Revision Level
	ATZ	Reset Device, forces a Status Message
	ATS8=n	Set Register 8 - Delay time in seconds. Default is 2 seconds.
5	ATS11=n	Set Register 11 - Time between DTMF codes in msec. Default is 95 msec.
	ATX3	Don't wait for Dial Tone before dialing Note: This will cause the Device to not wait for dial tone before initiating the call, but all dial modifiers are still in effect.
10	ATX4	Wait for Dial Tone before dialing (default)
	ATA	Answer
15	ATLn	Speaker Volume Control where n is in the range of 0 to 15.

In addition the device 120 supports these added, device specific commands:

20	AT+DTMF	This command will be given when the host sees the DTMF bit set in the Status byte. When given, the Device uploads all characters in its DTMF buffer using the DTMF Data Message. The Device will insert a Carriage Return (0x0D) character into its DTMF buffer when it detects the handset going on hook. The Device will maintain at least a 256 byte buffer per telephone line. As the buffer fills, the oldest characters will be overwritten.
25		
30	AT+LSn	This command sets the phone line. Valid values are 1 and 2. Any commands given after this command will apply to the set line. The Device powers up in a Line 1
35		

		setting.
5	AT+BURSTnn \bar{d} c	This command initiates the sending of a packet of data (DTMF codes). nn specifies the length of the data plus checksum (the first n indicates the high byte and second n indicates the lowbyte), \bar{d} is the data and c is the checksum.
10	AT+LINKnn \bar{d} c	This command initiates a link between two attached Devices. nn indicates the length of the send data plus checksum (the first n indicates the high byte and second n indicates the low byte), \bar{d} is the data and c is the checksum.
15	AT+BRESEND	This command is a signal from the shell that the destination workstation did not get a valid packet, probably due to a checksum error. This command prompts the destination Device to send a message to the origin Device on the other end.
20	AT+BCONFIRM \bar{c}	This command is a signal from the shell that the destination Device received a valid packet, confirmed by a proper checksum. This command prompts the destination Device to send a message to the origin Device on the other end. \bar{c} is the checksum.
25		
30	AT+M(x)(y)(1/0)	Matrix command (x) + (y) specify index into the matrix of Table 3. A (1) establishes the connection while a (0) breaks the connection.
	ATP11	Enter local mode, Line 1.
	ATP10	Enter normal mode, Line 1.
	ATP21	Enter local mode, Line 2.
35	ATP20	Enter normal mode, Line 2.

Some examples of the matrix commands are as follows:

Hold - music on hold;

Connect Lines - conference call; and

Speaker Phone - use speaker phone voice.

5 The device 120 responds to every AT command, except ATD
and ATZ, with a text string indicating the result of the
command. "Error" indicates that the device 120 was unable to
parse or execute the command. For data burst transfer, a
checksum error or the lack of a device 120 at the other end
10 of the line triggers a sending of a "BurstError" DLE message
to the origin host. "BurstReceived" DLE message indicates a
successful execution of the data burst transfer.

15 III. Data Burst Function

20 The data burst function will now be more fully explained
by reference to Figures 4-12. Referring now to Figure 4, the
User Interface (UI) layer will be described in a send
process. This layer interfaces the application software with
hardware/software that is used to implement the data burst
function. The flow shown in Figure 4 is a general or generic
user interface. A specific use of the data burst transfer
protocol, implemented in the UI layer, namely, the Business
Card Transfer application, will be described in conjunction
with Figure 13.

25 The burst function begins at a start state 402 and moves
to "startdata" function 404, which is at the manager layer
148 (Figure 2). The call to the function 404 initiates a
send of a data burst. Function 404 will be more fully
discussed in conjunction with Figure 5 hereinbelow.

30 Upon return from function 404, a test is made at a
decision state 406 to determine if a burst error flag was set
in function 404. A burst error indicates a fatal problem
that causes a discontinuation of the send process. If so,
the burst function moves to a state 408 wherein the
35 application software will either abort the send or restart

-31-

the send operation depending on the application. If there is no burst error, as determined by state 406, the burst function moves on to state 410 wherein a four byte tag and a stream of data are composed or made into a packet to be sent by the "senddata" function 412, which is at the manager layer. The tag is used to uniquely identify the type of information contained in the packet. The function 412 actually sends the data burst in a subpacket that is up to 512 bytes in size, which may or may not account for the entire packet size. Function 412 will be more fully described in conjunction with Figure 6.

Upon return from function 412, a test is made at a decision state 414 to determine if burst error was set in function 412. If so, the burst function moves to a state 408 wherein the application software will either abort or restart the send operation depending on the application. Note that upon receipt of a burst error message, an "enddata" function 422, described hereinbelow, does not need to be called. If there is no burst error, as determined by state 414, the burst function moves on to a decision state 416 wherein a determination of whether all the data of the current data packet has been sent. For example, the total data packet size may be 678 bytes, but only 512 bytes are sent in a subpacket at one time, so after the first send, 166 bytes still need to be sent.

In the Electronic Business Card feature of the present invention, a packet is defined as one database attribute relevant to a person, e.g., a business street address, or in other words, a group of data sent under the same tag or header. Therefore, when the data packet is not complete as determined by state 416, the burst function moves to state 418 wherein the next packet is accessed in preparation for the next call to the senddata function 412.

If one entire data packet has been sent, as determined by state 416, the burst function proceeds to a decision state

420 to determine if the application software has another packet to send. Most applications will send a plurality of data packets in one data burst. If the last data packet has not been sent, the burst function loops back to state 410 to
5 compose the next data packet. However, if the last packet has been sent, as determined by state 420, the burst function moves to the "enddata" function 422, which is at the manager layer. The call to the function 422 signals completion of one or more data bursts. Function 422 will be more fully
10 discussed in conjunction with Figure 7 hereinbelow.

Upon return from function 422, a test is made at a decision state 424 to determine if burst error was set in function 422. If so, the burst function moves to a state 408 wherein the application software will either abort the send
15 or restart the send operation depending on the application. If there is no burst error, as determined by state 424, the burst function moves on to an end state 426 and the user can again initiate a data burst.

To prevent the UI from freezing up during the process, calls to the "senddata" function 412 take place within a Window routine, which is message/interrupt driven. The Manager and Tools endeavor to return control to the UI layer as quickly as they can. During a data burst, functions for the line being used are disabled with the exception of a few,
20 e.g., hang up or flash.

Referring now to Figure 5, the "startdata" function 404 of the manager layer shown in Figure 4 will be described. Function 404 initiates the sending of data bursts and verifies the path from a origin workstation to a destination
30 workstation as shown in Figure 1. For example, workstation 110a may be the origin workstation and workstation 110b may be the destination workstation. The burst function begins at a start state 440 and moves to a "beginboxdata" function 442, which is at the tools layer. Function 442 creates a
35 beginning packet in a format such that the DTMF Interface

device 120 can begin transmission or respond. Function 442 will be explained more fully in conjunction with Figure 8.

5 Upon return from the function 442, the burst function determines at a decision state 444 whether the function 442 returned any errors, e.g., no response by the DTMF Interface
device 120 or a timeout. State 444 also includes a call to a function named "check2sum" that calculates and verifies the
checksum sent with the current packet. The checksum is as
previously described hereinabove. If an error is detected by
10 state 444, the burst function proceeds to a decision state 446 wherein a determination is made whether a predetermined threshold number of errors has been reached. The manager layer keeps track of the number of errors in a static variable within the manager. The number of errors is cleared
15 after each successful send of a packet to the destination host. If the number of errors in the static variable reaches the threshold number, a burst error flag is set and reported back at state 448 to the user interface (UI) layer (Figure 4) and control is returned to the UI layer through a return
20 state 450. If the preset maximum errors have not been reached as determined by state 446, the burst function loops back to state 442 to try sending the beginning packet again.

 If there are no transmission errors as determined by state 444, the burst function advances to state 452 and
25 initiates the dispatch of the packet from the origin DTMF Interface device 120a as shown in Figure 1 to the destination device 120b. The packet is sent via DTMF codes including a single byte at the end indicating checksum. At state 454, the destination device 120b receives the packet. Then at a
30 decision state 456, a determination is made by the device 120b whether the checksum is correct. This state includes a call to the substantial equivalent of the "check2sum" function, mentioned earlier, that calculates and verifies the checksum sent with the current packet.

35 If the checksum is not correct at state 456, the

-34-

destination interface device 120b returns an error at state 458 and the burst function moves back to state 446 to check if the preset error threshold has been reached. However, if the checksum is correct at state 456, the packet is sent from the destination device 120b to the destination workstation or host 110b at state 460. Then, at a decision state 462, a determination is made whether the checksum at the destination workstation 100b is correct, including a call to the "check2sum" function, mentioned earlier. If the checksum is not correct at state 462, the destination interface device 120b returns an error to the originating device 120a and thence to the originating host 110a at state 458 and the burst function moves back to state 446 to check if the preset error threshold has been reached. However, if the checksum is correct at state 462, a burst success message is sent by the destination device 120b to the originating device 120a and thence to the originating host 110a and a return is done at state 450. The burst success message is a confirmation to the originating workstation 110a of the successful sending of a packet. Until this message is received by the UI, subsequent calls to the "senddata" function 412 are not made.

The states 444 through 464 form a common set of steps that are also utilized in the functions 412 and 422. The description for these steps will not be repeated, but reference will be made back to states 444 to 464.

Referring now to Figure 6, the manager layer "senddata" function 412 shown in Figure 4 will be described. Function 412 sends a data burst from the origin workstation 110a to the destination workstation 110b as shown in Figure 1. The burst function begins at a start state 480 and moves to a state 482 wherein a check is made for valid data length and a valid pointer to the data buffer. Then at state 484, a determination is made whether the data packet needs to be split up. This determination is actually part of a

"splitdata" function 486. If the data length is greater than 512 bytes, the function 486 splits the burst packet into a piece small enough to send and stores the remainder in a dynamic-length buffer "rbcur". A dynamic length buffer does not have a fixed length assigned for it, but is dynamically allocated.

However, if the data length is not greater than 512 bytes, the burst function moves to "check2sum" function 488. Function 488 calculates a one-byte simple data packet checksum by adding the ASCII value of all characters in the string and then XORing the sum with the value 0xFF. In this specific instance of the "check2sum" function 488, the verification step is not done. Upon return from function 488, the burst function proceeds to state 490 wherein the current data packet being sent is buffered in a dynamic-length buffer "bcur" in case a resend operation is necessary. Resend will be more fully discussed hereinbelow.

Then the burst function moves to a "boxdata" function 492, which is at the tools layer. Function 492 transforms the data packet into a packet with a format that the DTMF Interface device 120 can understand. Function 492 will be explained more fully in conjunction with Figure 9. Upon return from the function 492, the burst function moves to decision state 494. States 494 through 514 are essentially the same as the states 444 through 464 of Figure 5. Therefore, to avoid redundancy, the description for these steps will not be repeated, but reference is made back to states 444 to 464. The return to the user interface layer (Figure 4) is at state 500.

Referring now to Figure 7, the manager layer "enddata" function 422 shown in Figure 4 will be described. Function 422 signals completion of the sending of packets from the origin workstation 110a to the destination workstation 110b as shown in Figure 1. The burst function begins at a start state 530 and moves to a "endboxdata" function 532, which is

at the tools layer. Function 532 creates an ending packet in a format that the DTMF Interface device 120 can understand. Function 532 will be explained more fully in conjunction with Figure 10. Upon return from the function 532, the burst function moves to decision state 534. States 534 through 554 are essentially the same as the states 444 through 464 of Figure 5. Therefore, to avoid redundancy, the description for these steps will not be repeated, but reference is made back to states 444 to 464. The return to the user interface layer (Figure 4) is at state 540.

Referring now to Figure 8, the tools layer "beginboxdata" function 442 shown in Figure 5 will be described. Function 442 sends an initial subpacket across the DTMF Interface device 120 (Figure 1) to signal that subsequent subpackets will be coming across the line. The burst function begins at a start state 570 and moves to a state 572 wherein a beginning packet is created in an AT+BURST format. This packet has the two length bytes equal to 2 in binary format, a beginning marker <cntrl-Q>, and the appropriate checksum. At state 574, this packet is sent to the origin DTMF Interface device 120a (using the same example origin device as earlier). At a decision state 576, a determination is made if the device 120a received a good packet. If so, a return is made at state 580 back to the function 404 (Figure 5). If not, a "no response" or error status is returned to the "startdata" function 404 followed by the return at state 580.

Referring now to Figure 9, the tools layer "boxdata" function 492 shown in Figure 6 will be described. Function 492 sends a subpacket across the DTMF Interface device 120 (Figure 1). This may be the only portion of the data needed. It will periodically be called to send data until the transmission is completed. The burst function begins at a start state 590 and moves to a state 592 wherein a data subpacket is created in an AT+BURST format. This subpacket

has the two length bytes, the data stored into a string to be sent, the appropriate checksum, and a carriage return (\r). At state 594, this subpacket is sent to the origin DTMF Interface device 120a (using the same example origin device as earlier). At a decision state 596, a determination is made if the device 120a received a good packet. If so, a return is made at state 600 back to the function 412 (Figure 6). If not, a "no response" or error status is returned to the "senddata" function 412 followed by the return at state 600.

Referring now to Figure 10, the tools layer "endboxdata" function 532 shown in Figure 7 will be described. Function 532 sends an ending or final subpacket across the DTMF Interface device 120 (Figure 1) to signal that no more subpackets for the current packet or database field are forthcoming. The burst function begins at a start state 610 and moves to a state 612 wherein an ending packet is created in an AT+BURST format. This packet has the two length bytes equal to 2 in binary format, an ending marker <cntrl-S>, and the appropriate checksum. At state 614, this packet is sent to the origin DTMF Interface device 120a (using the same example origin device as earlier). At a decision state 616, a determination is made if the device 120a received a good packet. If so, a return is made at state 620 back to the function 422 (Figure 7). If not, a "no response" or error status is returned to the "enddata" function 422 followed by the return at state 620.

Referring now to Figure 11, a receive process as performed at the tools layer will be described. This process takes place when the destination workstation 110b as shown in Figure 1 (using the example destination workstation as earlier) receives a packet. The burst function begins at a start state 650 and moves to a state 652 wherein a data subpacket is received at a communication port receive buffer (not shown) of the destination workstation 110b. Then, at

state 654, the tools layer receives a "WM_COMMNOTIFY" message from Windows as a notification that a message has been received by the receive buffer. The burst function moves to "tools_parse" function 656 which includes a state machine to
5 determine the packet message type. Function 656 also copies the data in the receive buffer to a dynamically allocated memory block to store data in a string format. Function 656 is called by the tool layer whenever there are characters in the buffer. It continues until all of the characters are
10 processed. However, it does not rely on all of the data to be present in the buffer when processing begins. Therefore, messages can span two or more calls to the parser. Function 656 is set up as a state machine to look for the <DLE> character as the start of commands or messages from the
15 device 120 or <CR> character as the start of an alphanumeric response.

A decision state 658 then determines if the subpacket contains a burst message. If not, the burst function moves to state 660 and waits for the next data subpacket. If the
20 subpacket contains a burst message as determined by state 658, the burst function proceeds to state 662 wherein the data string is sent to the manager for parsing into a bMsg_st structure. The bMsg_st is a data structure within the memory of the workstation 110 which temporarily stores information
25 associated with the data burst. This structure includes a four byte tag to identify the data type, the length of data in bytes, and a union of three pointers to the data in three different formats. A "telcallback" function 664, which is at the manager layer 148 (Figure 2), is then called. The
30 function 664 receives messages from the tools layer 150, processes them, and returns the results to the UI layer 146. The function 664 will be further described in conjunction with Figure 12.

Upon return from the function 664, a decision state 666
35 determines if the manager layer called the "resend" function

708. If so, the tools layer "resend" function 668 is called. Function 668 sends a message to the destination device 120b to issue a resend command, which requests a resend of the last subpacket, to the device 120a at the other end of the connection. Subsequently, the burst function proceeds back to state 660 to wait for the resent subpacket to be received. If state 666 is false, the burst function moves to a "confirm" function 670 at the tools layer. The function 670 sends a message to the destination device 120b to issue a confirm command to the origin device 120a at the other end of the connection. Upon return from the function 670, the burst function proceeds to state 672 wherein the memory allocated as part of the "tools_parse" function 656 is deallocated. The memory is not needed since the data is copied by the manager at function 664. The burst function then moves to a decision state 674 to determine if the user wishes to quit the receive process. If so, the burst function moves to a state 676 and exits the burst process. If not, the burst function moves to state 660 and waits for the next data burst.

Referring now to Figure 12, the manager layer "telcallback" function 664 shown in Figure 11 will be described. Function 664 is called by the tools layer to parse the data string sent by tools into the bMsg_st structure previously mentioned. The burst function begins at a start state 690 and moves to a decision state 692 wherein a determination is made whether the string has a beginning marker (<cntrl-Q> in the presently preferred embodiment). If so, the burst function moves to state 694 wherein a "TM_BURSTACT" message is sent to the UI, followed by a return to the tools layer (Figure 11) at state 696. If the decision state 692 is false, the burst function moves to a decision state 698 wherein a determination is made whether the string has a ending marker (<cntrl-S> in the presently preferred embodiment). If so, the burst function moves to state 700

wherein a "TM_BURSTNOTACT" message is sent to the UI, followed by a return to the tools layer (Figure 11) at state 696. If the decision state 698 is false, the burst function proceeds to a "parsedata" function 702. The function 702 takes the data string, parses it, and fills in the bMsg_st structure with tag and data in preparation to be passed up to the UI layer. Upon return from the function 702, a "check2sum" function 704 calculates and verifies the checksum. The checksum is as previously described. If the checksum is not correct as determined at a decision state 706, a manager layer "resend" function 708 is called, which will call the "resend" function 668 at the tools layer (Figure 11). If the checksum is correct at state 706, the burst function proceeds to a decision state 710 to determine if the data is complete by checking a three byte marker or continuation string at the end of the subpacket. This check is actually part of a "unitedata" function 712. If the data is not complete, function 712 receives the incoming data subpacket and buffers it until the remaining data arrives in a subsequent subpacket. The return at state 696 is done if the data is not complete.

When the remaining subpackets arrive, the function 712 reunifies the data into a complete packet within the bMsg_st structure and then sends the structure to the UI at state 714. A "TM_BURST" message accompanies the structure, and then a manager layer "confirm" function 716 is called, which will call the "confirm" function 670 at the tools layer (Figure 11). Finally, a return at state 696 to the function 670 is done.

Software for the databurst function is, in a presently preferred embodiment, written in the "C++" language. The software described herein, some of which is listed in the attached Microfiche Appendix was translated from the source code into object code using the Borland "C++" compiler, version 3.1. Sections titled "mgrburst.txt" and

"tlburst.txt" are included in the Microfiche Appendix. Nonetheless, one skilled in the technology will recognize that the steps in the accompanying flowcharts can be implemented by using a number of different languages, language translators, computers and circuitries.

The manager layer functions as shown in the attached Microfiche Appendix are as follows: StartData, SendData, EndData, _check2sum, _resend, _confirm, _parseData, _splitData, _uniteData, TelCallBack. The tools layer functions as shown in the attached Microfiche Appendix are as follows: BeginBoxData, BoxData, EndBoxData, Resend, Confirm, TOOLS_Parse.

IV. Business Card Transfer Application

One or more applications programs in the user interface layer, available as part of the Windows dynamic link library (DLL), may be included under an applications module, e.g., telephone module, which is also part of the DLL. These are stand-alone programs that can be accessed by other functions, modules, or executable files.

The present invention includes an application called *Business Card Transfer*. This function transmits business card information from one workstation to another via the phone lines using the data burst function. The information is automatically formatted in the shell database, and a destination user is given the option of saving, modifying or destroying the entry. The Business Card Transfer function easily automates the formerly slow and error prone process wherein the user manually typed-in database entries.

Figures 13a and 13b are a flowchart which, in conjunction with the screen displays of Figures 14 to 20 illustrate one presently preferred embodiment of the Business Card Transfer (BCT) function software which transmits data via the system 100, and is a part of the present invention. It should be noted that, unless specified otherwise, the

operation described with reference to the flowchart of Figure 13, is within one of the workstations 110, e.g., workstation 110a, and input and output functions described therein are performed via the input/output devices 112-118 which communicate directly with the workstation 110a.

Referring to Figure 13, the BCT function, begins at a start state 740. In one presently preferred embodiment, BCT may be executed from Microsoft Windows® 3.1 or higher software running on the Microsoft MS-DOS operating system, preferably version 4.1 or higher. The Windows software may be run from the RAM memory within any one of the workstations 110. The shell software of the system 100 is normally initiated at Windows start-up, although it may also be initiated from the Windows Program Manager, or the Windows File Manager, for example, at a state 742. In any case, once the shell is open, the user selects which module, e.g., telephone, to run by clicking on the icon for the module. Typically, the user activates functions on a menu or dialog box by the "point and click" method associated with pointer devices such as the mouse 114. The user simply positions the cursor (not shown) on the video display 112 over the desired icon (or button in other instances) and depresses a button on the mouse 114. Of course, other indication methods may be used, such as a keyboard or trackball.

When the phone icon has been selected, a display screen as exemplified by Figure 14 is seen on the video display 112 of the workstation 110a. The user can then click on the Dial button 850 which retrieves a phone number list from which a person to be called is selected. At state 744, once the person is selected, the call is initiated. A display screen exemplifying the information for an outgoing call on line 1 is shown by Figure 15. At this point, the user at the origin end can begin voice communication with the user at the destination end as in a typical telephone call. Voice communication is possible throughout the flow shown in Figure

13.

At a decision state 746, a determination is made whether the send process or the receive process is to be executed on the user workstation 110. The origin workstation 110a executes the send process while the destination workstation 110b executes the receive process. Of course, either workstation 110 may transmit or receive data. If the send process is to be executed, the burst function moves to a state 748 wherein the user opens the phone menu and selects, for this example, the Initiate Business Card Transfer item 860 as illustrated by the screen display of Figure 16. This selection is done, for example, by moving the mouse to that line in the phone menu, or by using the keyboard down cursor key.

The burst function proceeds to state 750 wherein the user selects or deselects data attributes that are to be sent to the destination workstation 110b as illustrated by the screen display of Figure 17. The user can choose all information 870, all business information 872, all home information 874, or subsets of the business or home information. The subset is made by selecting (by clicking on the square box next to the attribute name, or others means) the desired attributes, e.g., company name 876 or fax number 878, from the business or home information. In the example screen display of Figure 17, business information attribute 880 is darkened to indicate that it has been selected. When the attributes are properly selected, the user clicks on the "OK" button 882. The send process then essentially executes the general User Interface flow of Figure 4.

Each attribute may be composed of one or more data fields. A list of potential data fields, including a tag and the associated data, is shown in TABLE 4.

TABLE 4

	<u>Tag</u>	<u>Data</u>
35	FINA	First Name

-44-

	MINA	Middle Name
	LANA	Last Name
5	HST1	Home Street 1
	HST2	Home Street 2
	HCTY	Home City
	HSTT	Home State
	HZIP	Home Zip-Code
10	CPNA	Company Name
	PSTN	Position
	OST1	Office Street 1
	OST2	Office Street 2
	OCTY	Office City
15	OSTT	Office State
	OZIP	Office Zip-Code
	HPHN	Home Phone
	OPHN	Office Phone
20	CPHN	Car Phone
	FAXN	FAX Phone
	NOT1	Note 1
25	NOT2	Note 2

The general User Interface (UI) flow is summarized by the states 752, 754, and 756. At state 752, the UI initiates the burst by calling the "startdata" function 404 (Figure 4). The UI then moves to state 754 and iterates through the data by calling the "senddata" function 412 for each checked attribute which ultimately becomes a packet. At state 756, the UI calls the "enddata" function 422 to signal the end of the burst. The flow then proceeds from state 756 through the off page connector A 758 on Figure 13a to the off page connector A 758 on Figure 13b and then to state 760.

At state 760, the burst function displays the phone module screen as previously illustrated by Figure 15. Note that data is transferred during states 752 to 756 and voice communication over the telephone line is not practical (but is possible) during that time due to the DTMF tones being transmitted. Also of note is that voice communication during the data transfer time will not cause data transmission errors or terminate the data transfer. Good voice

communication can be resumed after state 756. The "HANG UP" button is selected when the user desires to terminate the voice communication, and a screen display as previously shown by Figure 14 is seen on the video display 112. At a decision state 762, the user either quits the shell and returns to the Windows environment at state 764, or loops back through the off page connector C 766 of Figure 13b to the off page connector C 766 of Figure 13a and subsequently to state 744.

If a receive process is to be executed as determined at state 746, the person at the origin end of the connection initiates a Business Card Transfer at state 780. The following description then applies to operation in the destination workstation 110b. At state 782, a structure to temporarily hold a database entry is set up by the burst function. This structure receives the attributes sent over with each call of the "senddata" function 412. Associated with the structure is unique identification (UID) information which includes a serial number of the origin device 120 and other information that is used to identify the structure. This UID information is indicative of an individual entity such as a person, company, or organization. After the structure is set up at state 782, the burst function moves to a decision state 784 to determine if the burst contains an attribute from a "senddata" function call or an endburst marker from an "enddata" function call. If the burst contains an attribute, the burst function moves to state 786 wherein the attribute is placed into the structure set up at state 782, followed by a loop back to state 784. This loop continues until an endburst is detected by state 784 and the burst function moves to state 788. At state 788, the structure that has been filled by state 786 is presented to a display routine. The burst function then proceeds through the off page connector B 790 to Figure 13b and further to a decision state 792 wherein the UID of the structure filled by state 786 is compared to the UIDs of entries in the shell

database. If the UID of the structure matches a UID in the database, the burst function proceeds to state 794 and displays the structure as illustrated by the screen display of Figure 18. Only the attributes selected by the user at the origin end are filled in. A set of four buttons 890, 892, 894, and 896 which correspond to states 796, 798, 800, and 802, respectively, are seen at the bottom of Figure 18.

If the user selects button 890, the burst function moves to state 796 and toggles between displaying the structure and the original database entry with the matching UID. The resultant screen display is illustrated by Figure 20. If the user selects button 892, the burst function moves to state 798 and saves the structure in the shell database as a new entry but gives it a new UID so that it does not save over the original entry. If the user selects button 894, the burst function moves to state 800 and does not save the structure. If the user selects button 896, the burst function moves to state 802 and saves the structure as an entry in the shell database. If the UID already exists in the database, the structure overwrites the original entry. After completion of state 798, 800, or 802, the burst function moves to state 760 and displays the phone module screen as previously illustrated by Figure 15.

If the UID of the structure does not match a UID in the database, i.e., the structure will be a new entry if saved, the burst function proceeds to state 804 and displays the structure as illustrated by the screen display of Figure 19. Only the attributes selected by the user at the origin end are filled in. A set of two buttons 900 and 902 which correspond to states 800 and 802, respectively, are seen at the bottom of Figure 19. States 800, 802, and further states of Figure 13b have been previously described.

While the above detailed description has shown, described and pointed out the fundamental novel features of the invention as applied to various embodiments, it will be

understood that various omissions and substitutions and changes in the form and details of the device illustrated may be made by those skilled in the art, without departing from the spirit of the invention.

WHAT IS CLAIMED IS:

1. In a telephone network, a data burst system, comprising:
 - a first workstation;
 - 5 first means for voice communication across the telephone network;
 - a first interface device connected to the first workstation, the first voice communication means and the telephone network, the first interface device including
 - 10 means for encoding data into DTMF tones and means for decoding DTMF tones into data;
 - a second workstation;
 - second means for voice communication across the telephone network;
 - 15 a second interface device connected to the second workstation, the second voice communication means and the telephone network, the second interface device including means for encoding data into DTMF tones and means for decoding DTMF tones into data;
 - 20 wherein voice communication between the first and second voice communication means and data communication between the first and second workstations can occur during a single telephone connection across the telephone network.
- 25 2. In the system defined in Claim 1, a method for transferring unique identification information indicative of an entity, comprising the steps of:
 - calling from the first voice communication means to the second voice communication means;
 - 30 selecting unique identification information stored on one of the workstations; and
 - sending the selected data to the other workstation.
3. The method defined in Claim 2, wherein the unique identification information includes an entity name.
- 35 4. The method defined in Claim 3, wherein the entity name

is a personal name.

5. The method defined in Claim 2, wherein the unique identification information includes a postal address.

5 6. The method defined in Claim 2, additionally comprising the steps of:

receiving the selected data at the other workstation; and

10 storing the selected data in a database at the other workstation.

7. The system defined in Claim 1, wherein the workstation includes a pointer device.

8. The system defined in Claim 1, wherein the first means for voice communication is a telephone.

15 9. The system defined in Claim 1, wherein data comprises ASCII data.

10. In a telephone network, a method of communicating data and voice signals between a first and second data burst system, each data burst system having means for voice communication, comprising the steps of:

establishing a telephone connection between the first and second systems;

communicating voice signals from one system to the other system;

25 translating data in the first system into a set of tones;

transmitting the tones from the first system to the second system; and

30 translating the tones at the second system into data.

11. The method defined in Claim 10, wherein each data burst system includes a computer having a processor and a memory.

12. The method defined in Claim 10, wherein each tone comprises a DTMF tone.

35 13. The method defined in Claim 10, wherein the data

comprises identification information indicative of an entity.

14. A data burst system, comprising:

means for processing and storing data;

means for converting data stored in the processing

5 means into converted data;

means for transmitting and receiving signals across
a switched telephone network wherein the signals include
voice signals and generated signals indicative of the
converted data;

10 wherein the voice signals and generated signals are
selectively switched in a single telephone connection.

15. The system defined in Claim 14, wherein the means for
transmitting and receiving includes means for voice
communication.

15 16. The system defined in Claim 14, wherein the generated
signals comprise tones.

17. The system defined in Claim 14, wherein the data
includes identification information indicative of an entity.

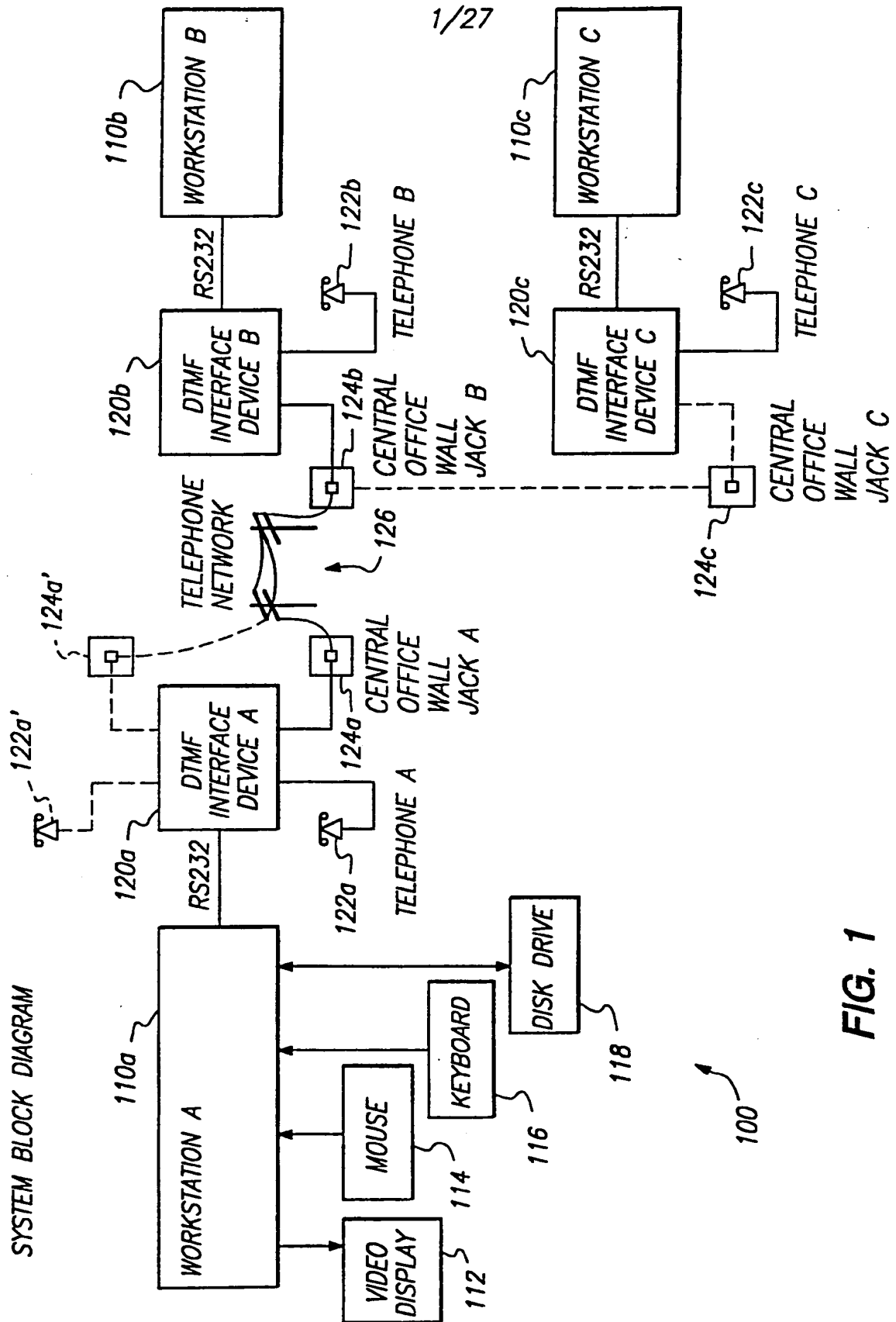


FIG. 1

2/27

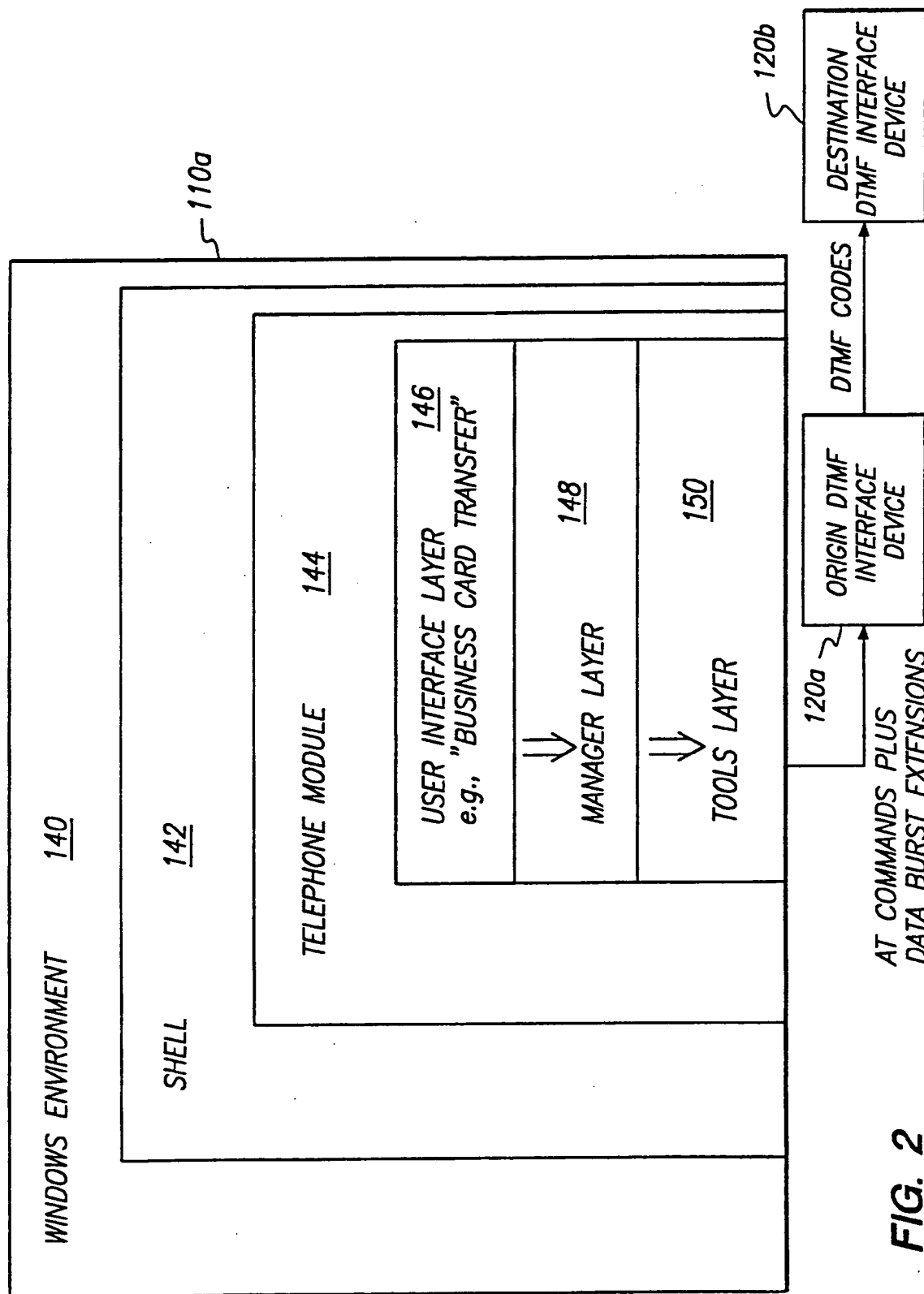


FIG. 2

SUBSTITUTE SHEET

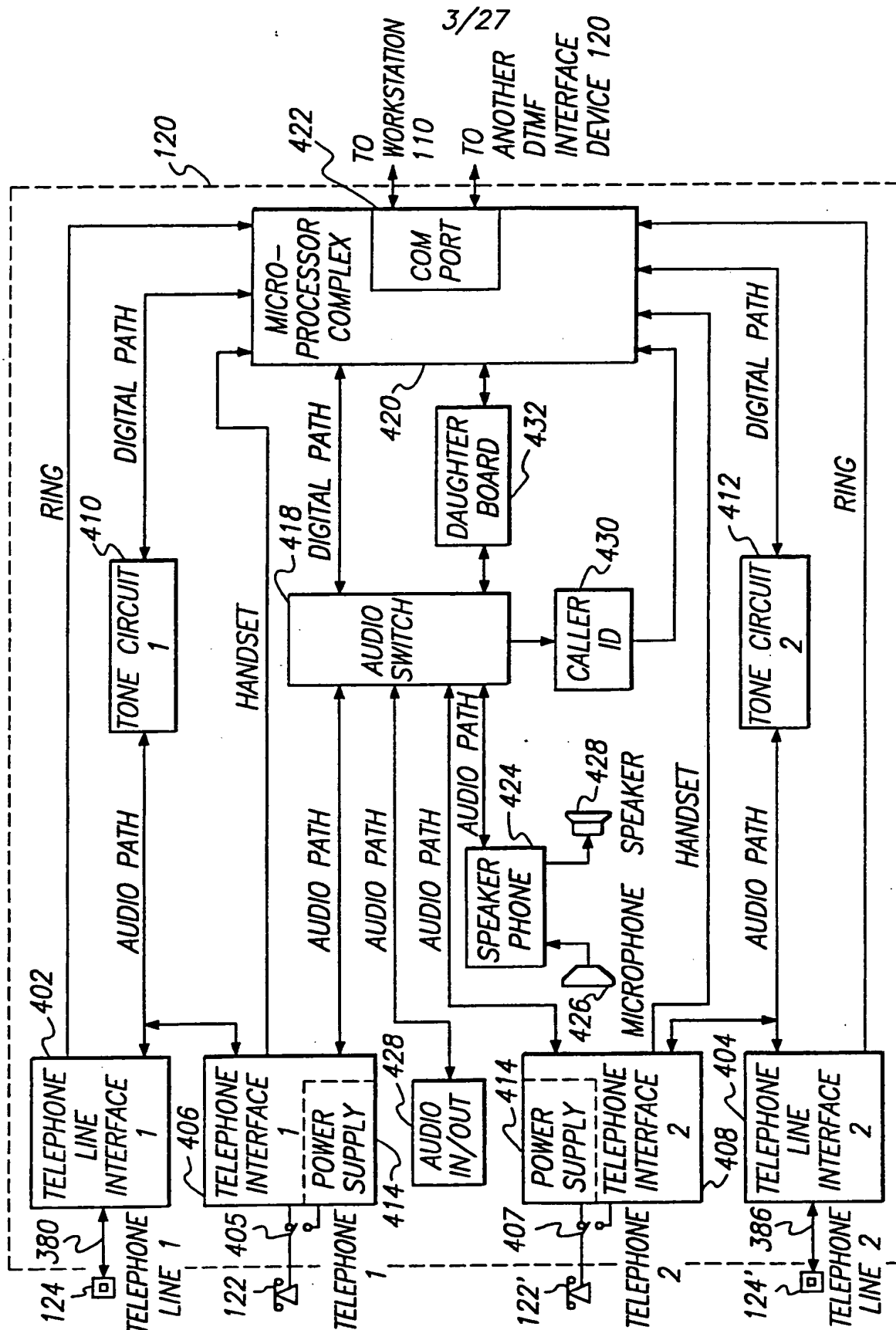


FIG. 3a

SUBSTITUTE SHEET

4/27

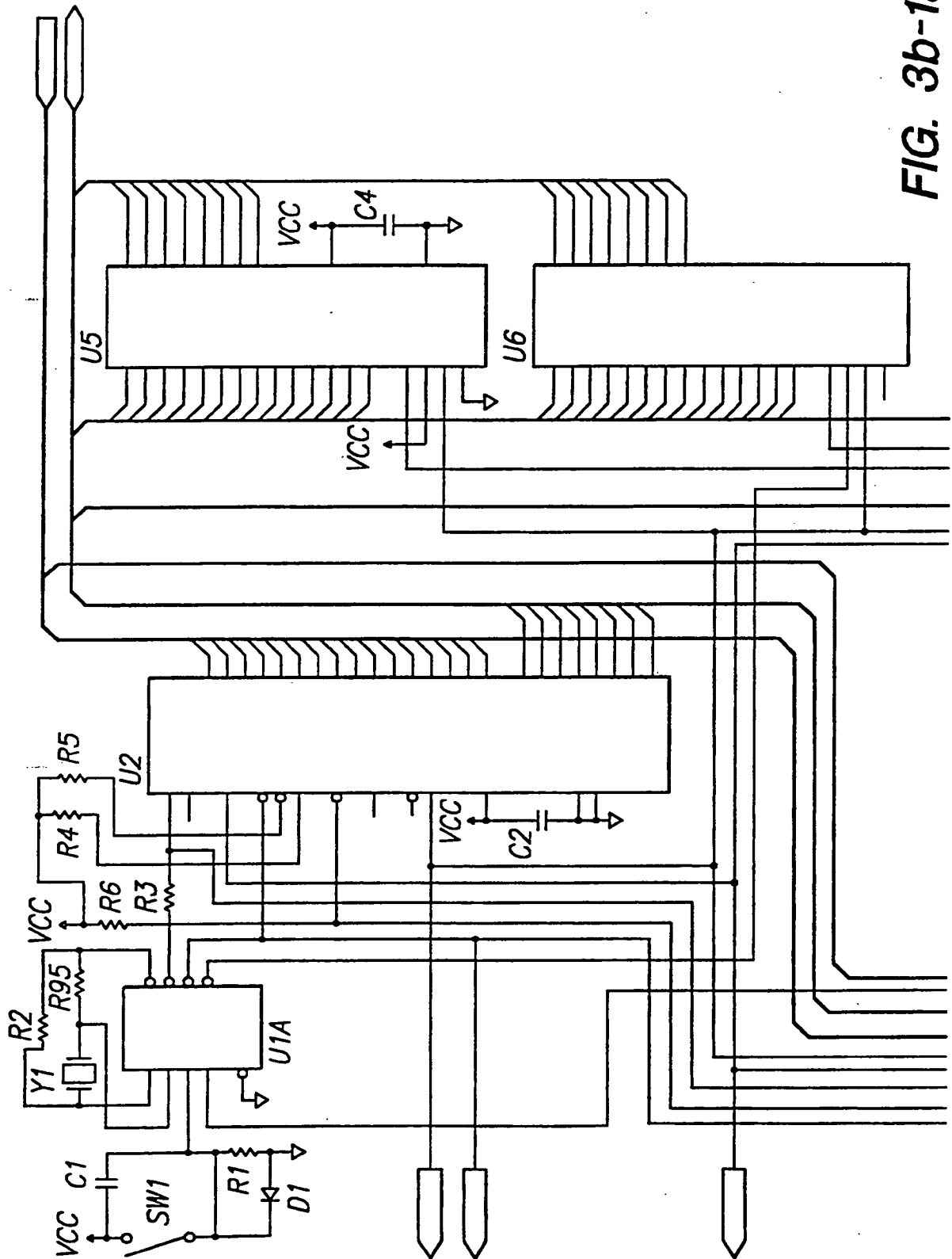


FIG. 3b-1a

SUBSTITUTE SHEET

5/27

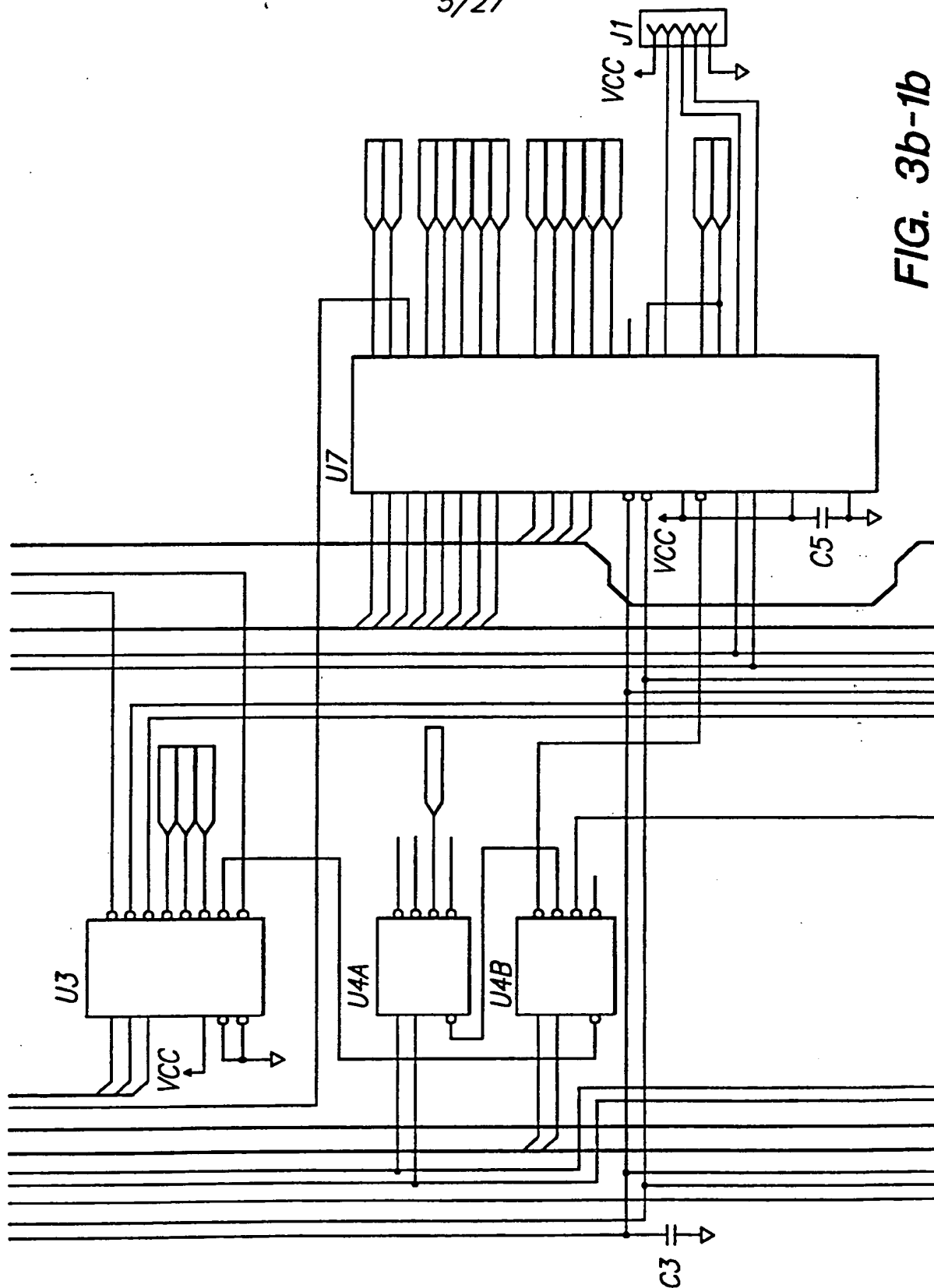


FIG. 3b-1b

SUBSTITUTE SHEET

6/27

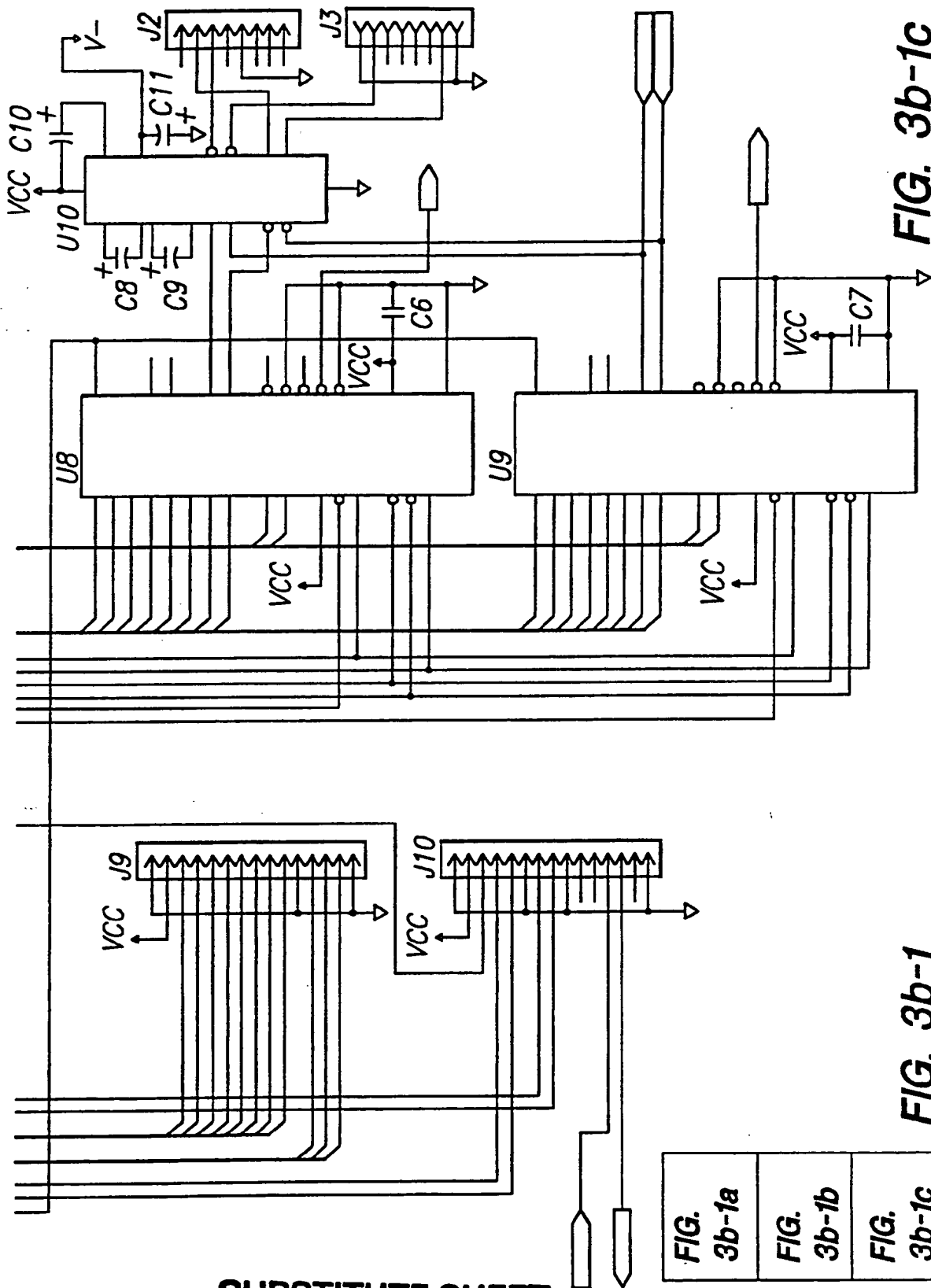


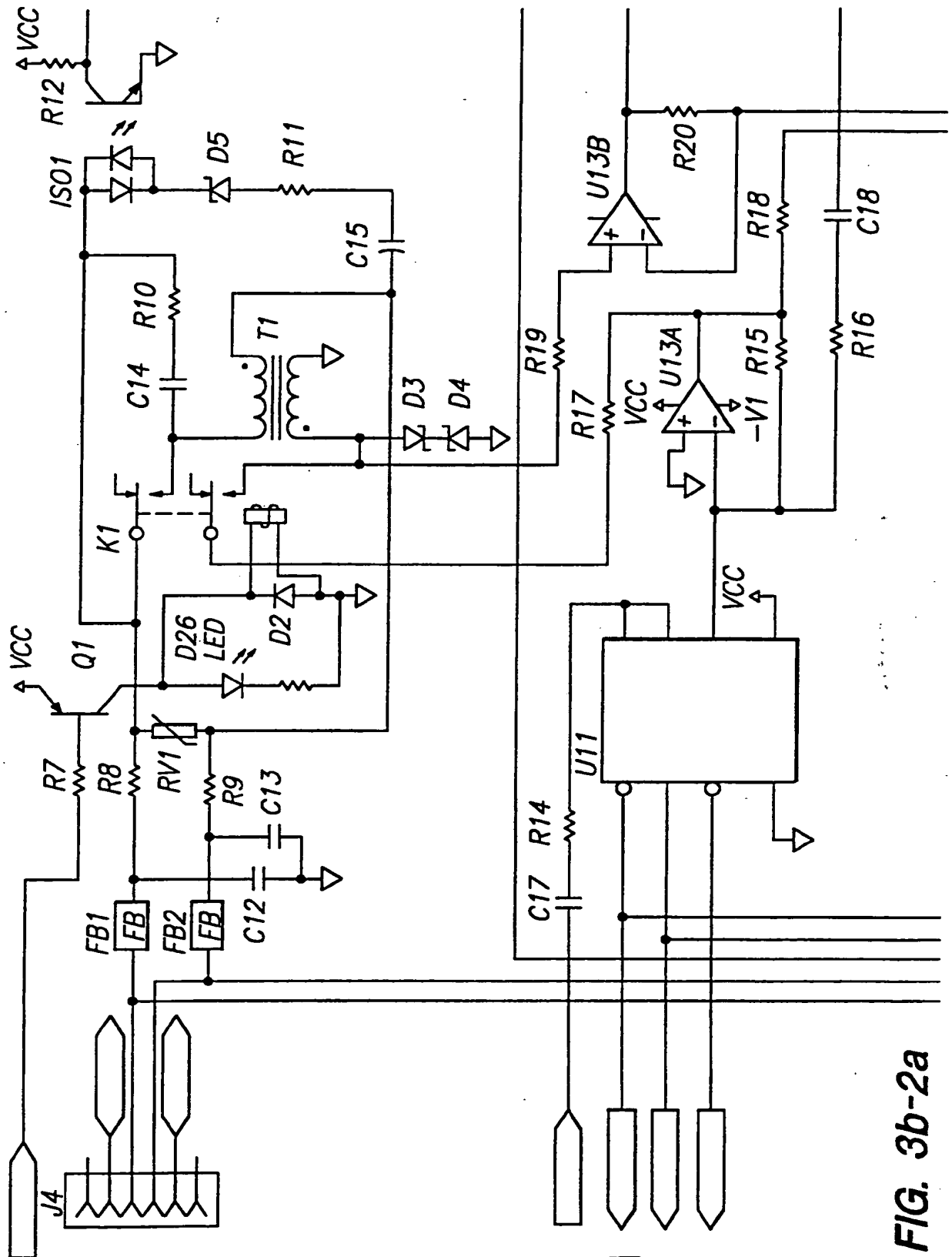
FIG. 3b-1c

FIG. 3b-1

FIG. 3b-1a	FIG. 3b-1b	FIG. 3b-1c
---------------	---------------	---------------

SUBSTITUTE SHEET

7/27



SUBSTITUTE SHEET

FIG. 3b-2a

8/27

FIG. 3b-2a	FIG. 3b-2b
FIG. 3b-2c	

FIG. 3b-2

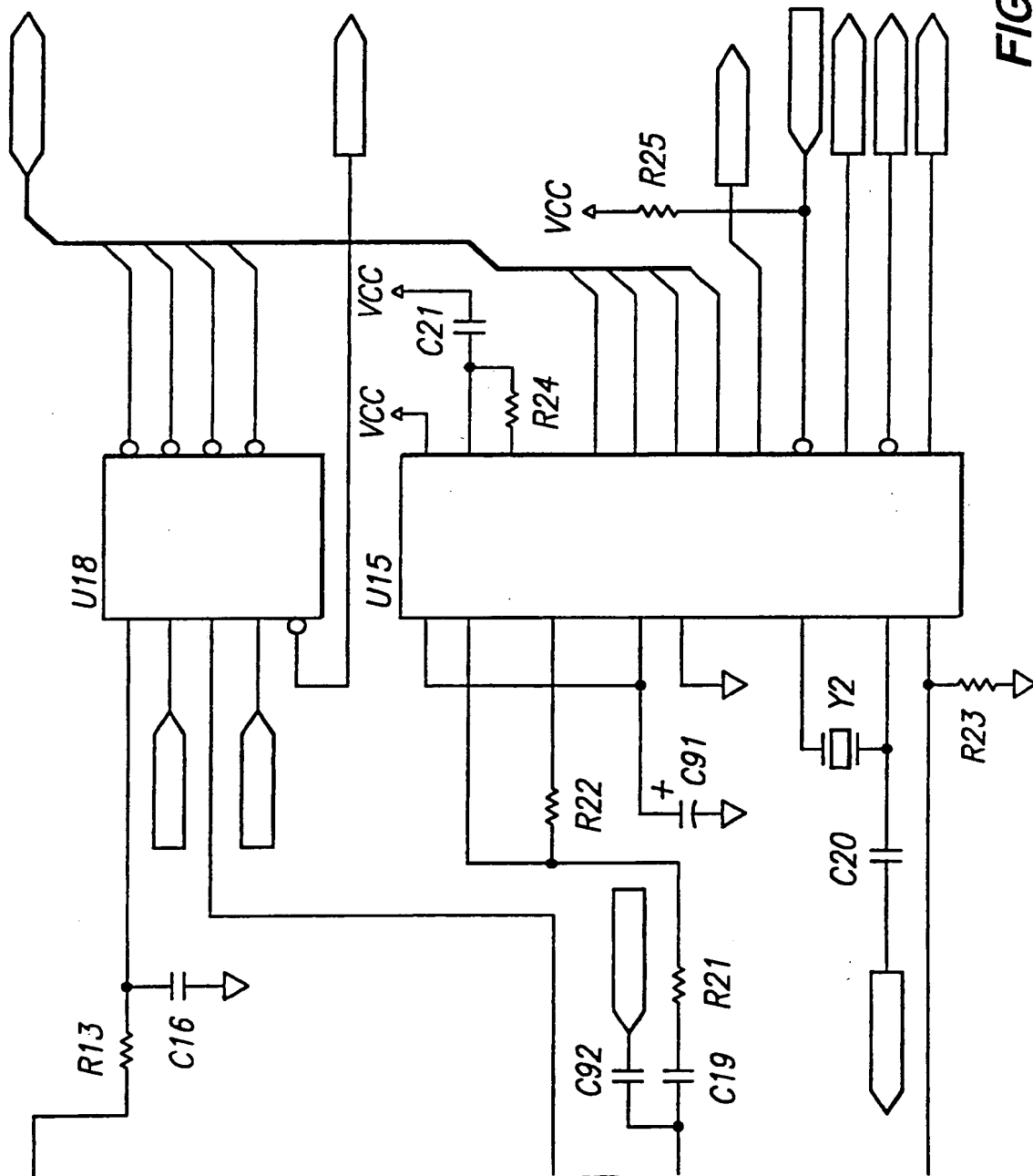
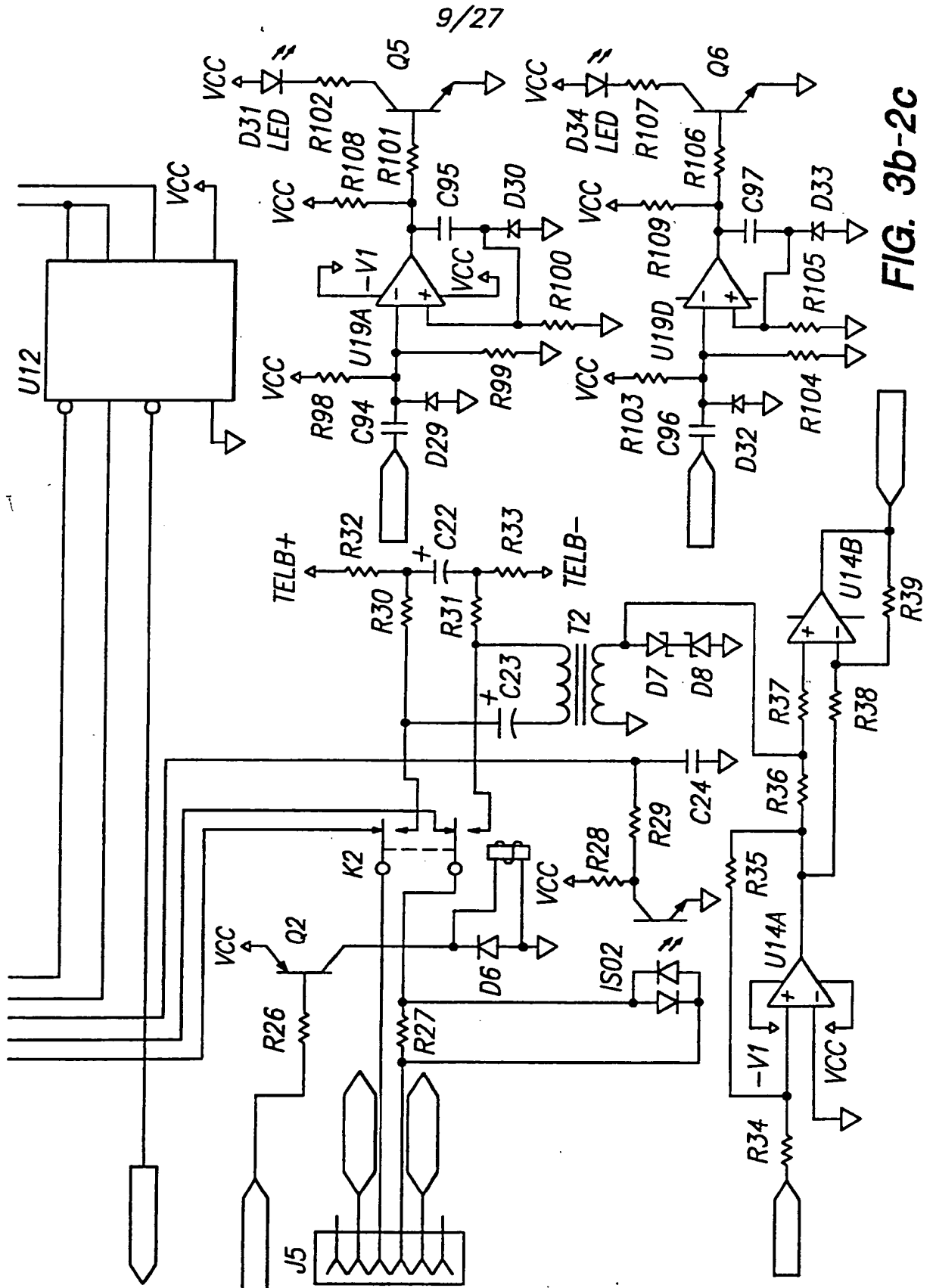


FIG. 3b-2b

SUBSTITUTE SHEET



SUBSTITUTE SHEET

10/27

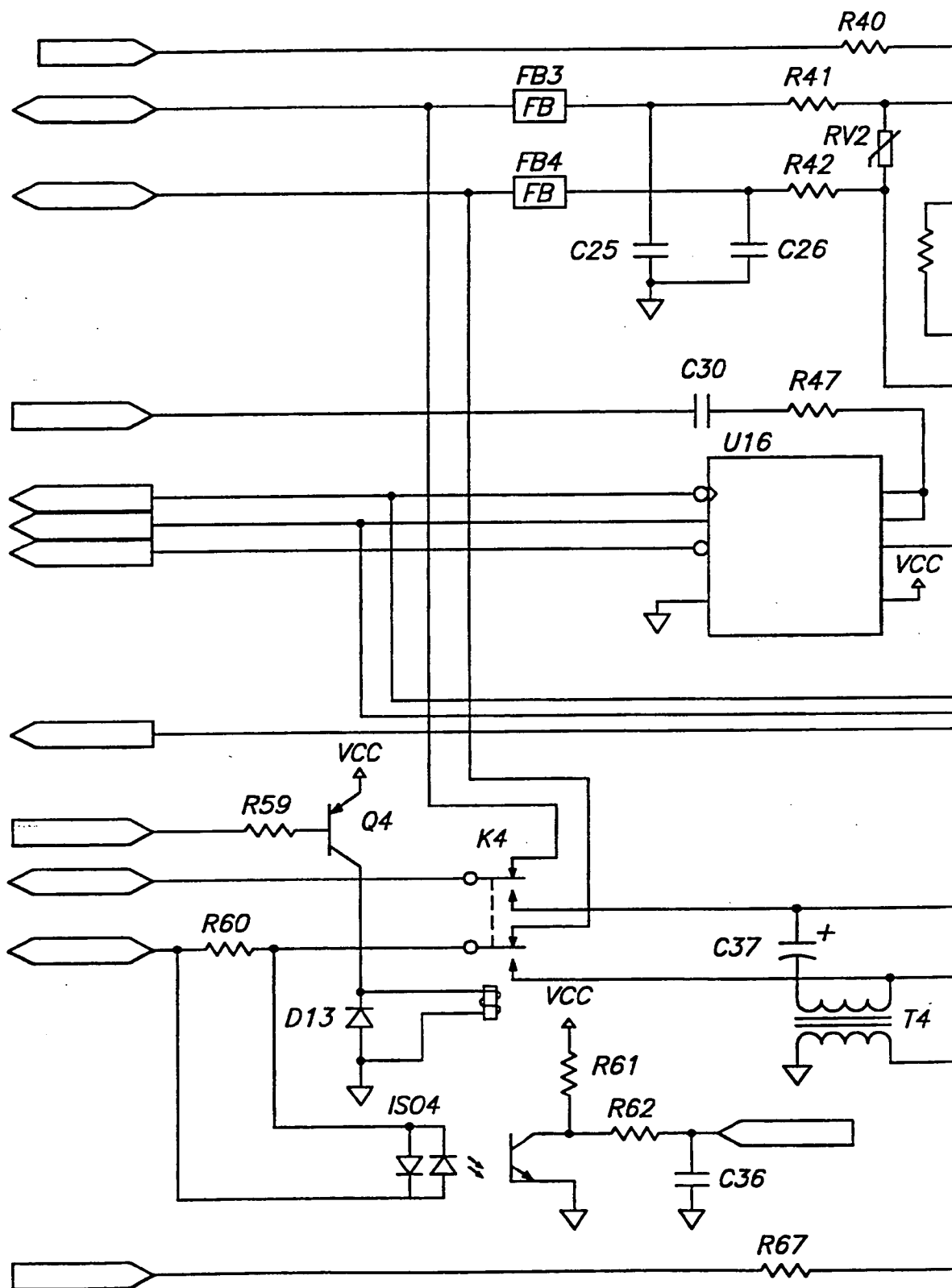


FIG. 3b-3a
SUBSTITUTE SHEET

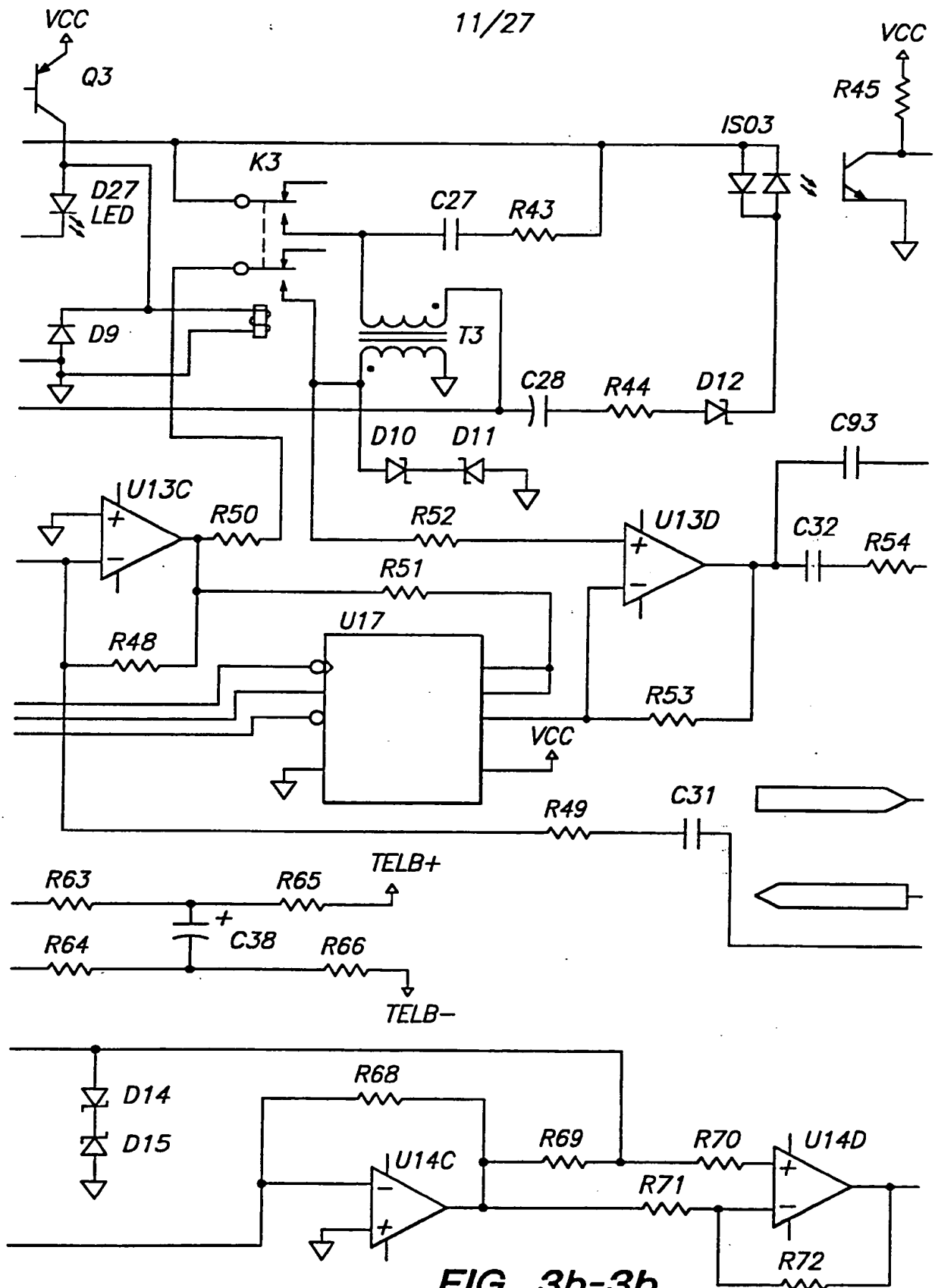


FIG. 3b-3b
SUBSTITUTE SHEET

12/27

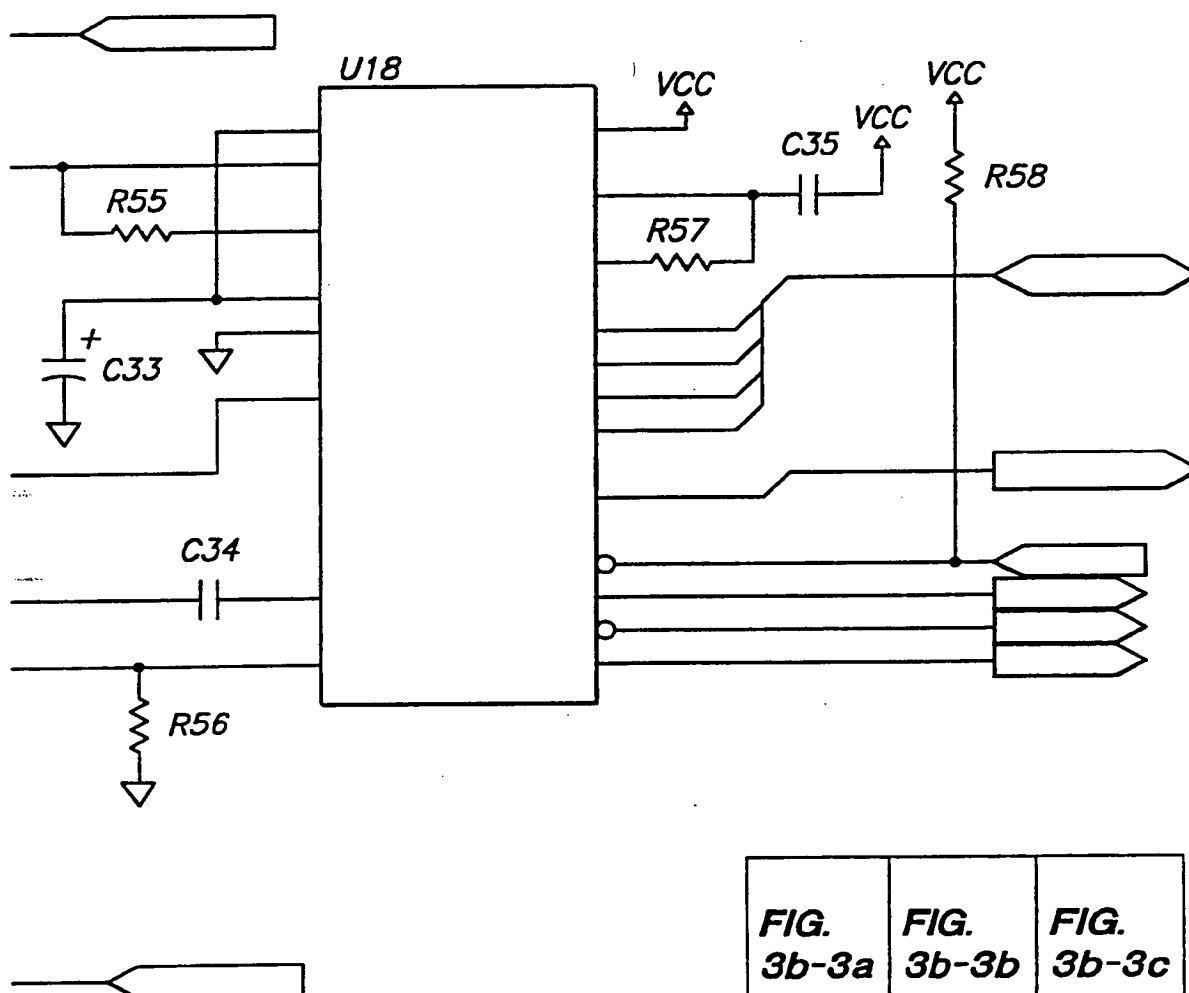
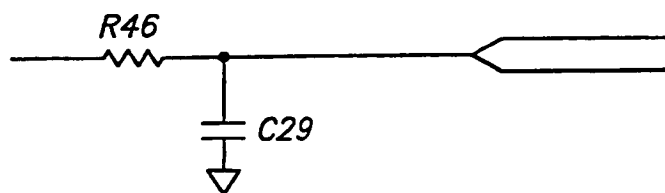


FIG. 3b-3c

SUBSTITUTE SHEET

FIG. 3b-3a	FIG. 3b-3b	FIG. 3b-3c

FIG. 3b-3

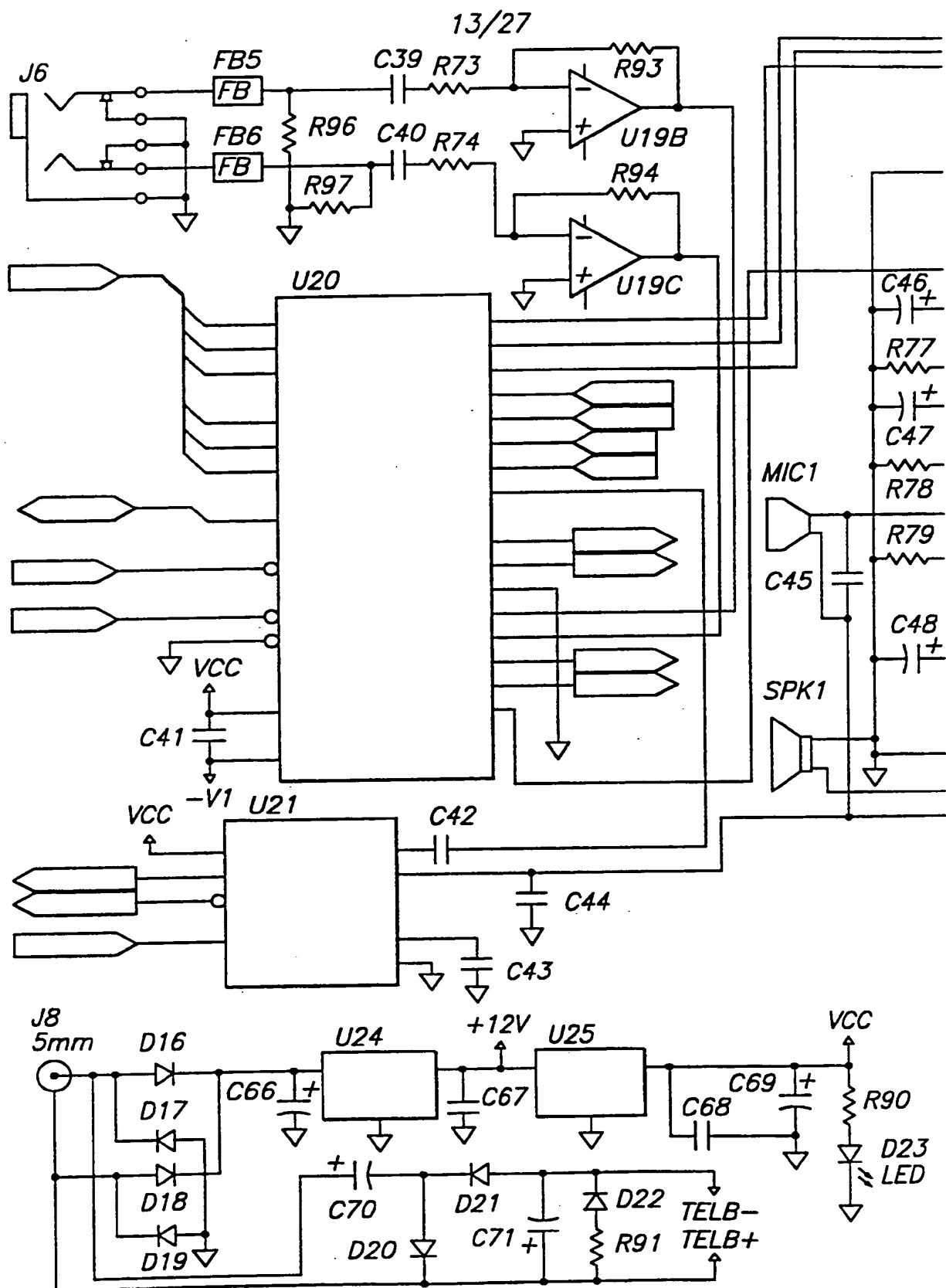


FIG. 3b-4a

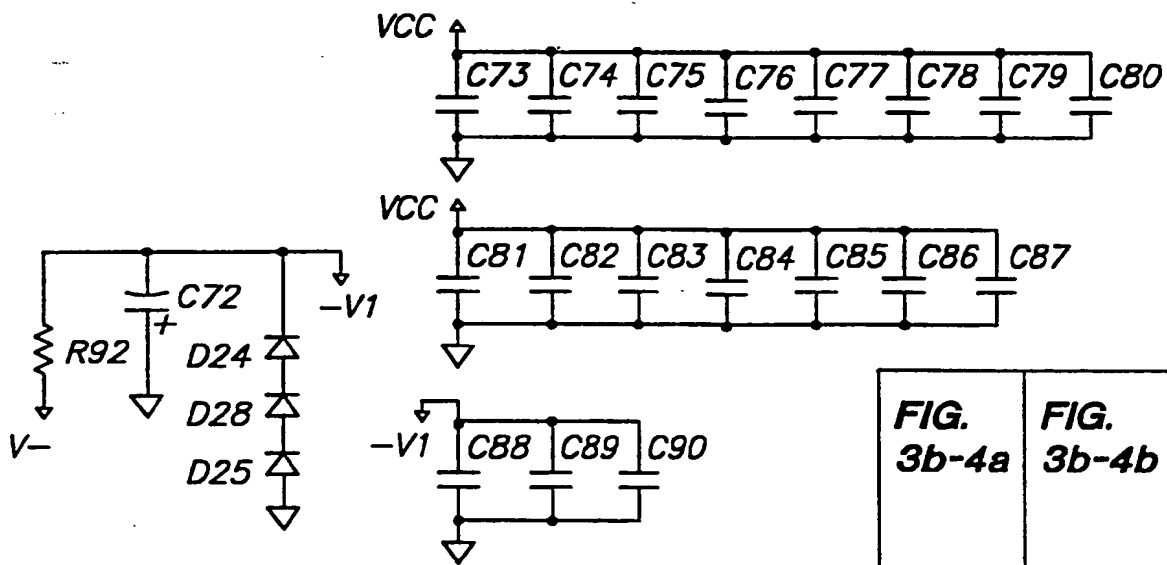
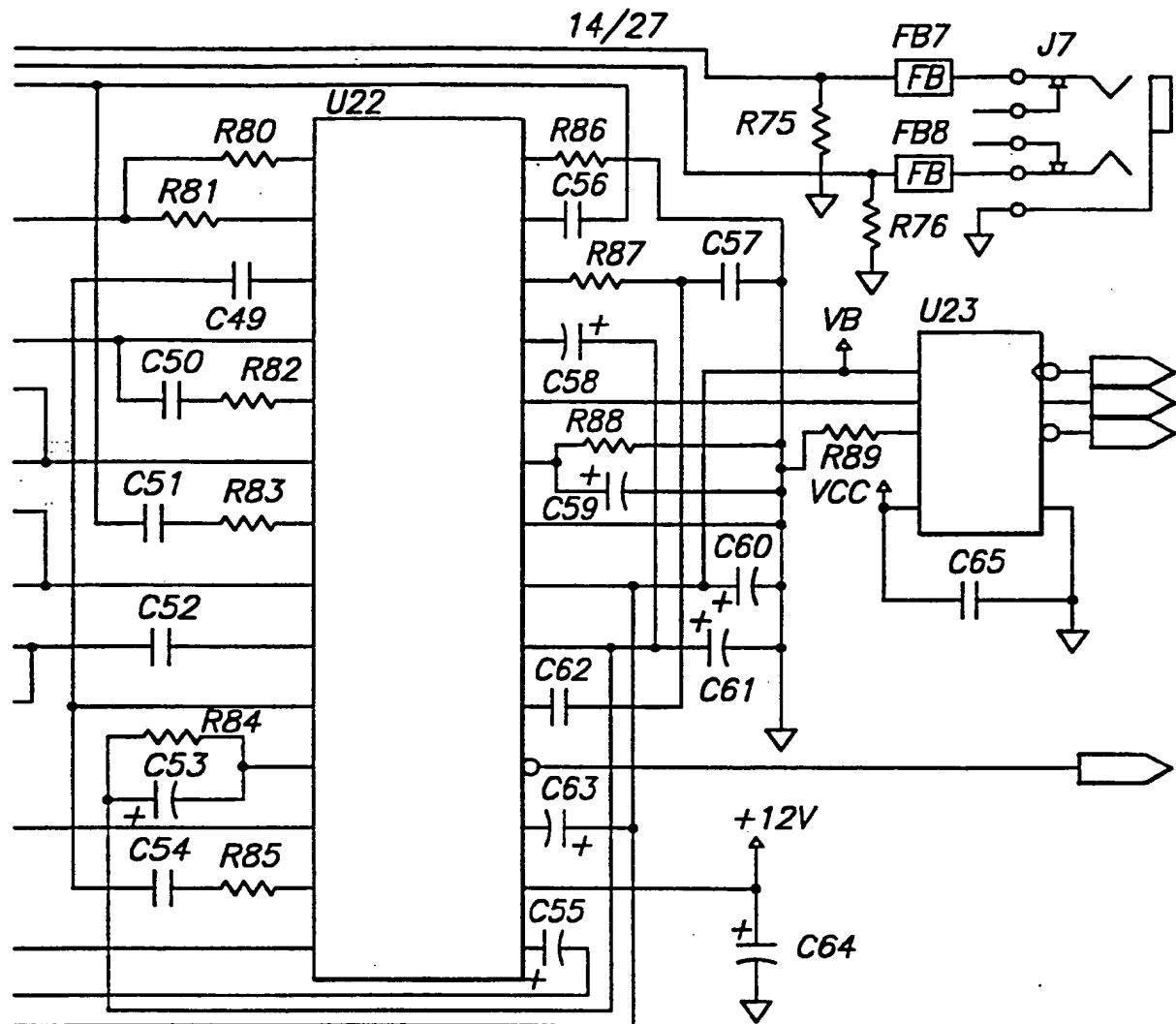


FIG. 3b-4b

FIG. 3b-4a	FIG. 3b-4b
-----------------------------	-----------------------------

FIG. 3b-4

SUBSTITUTE SHEET

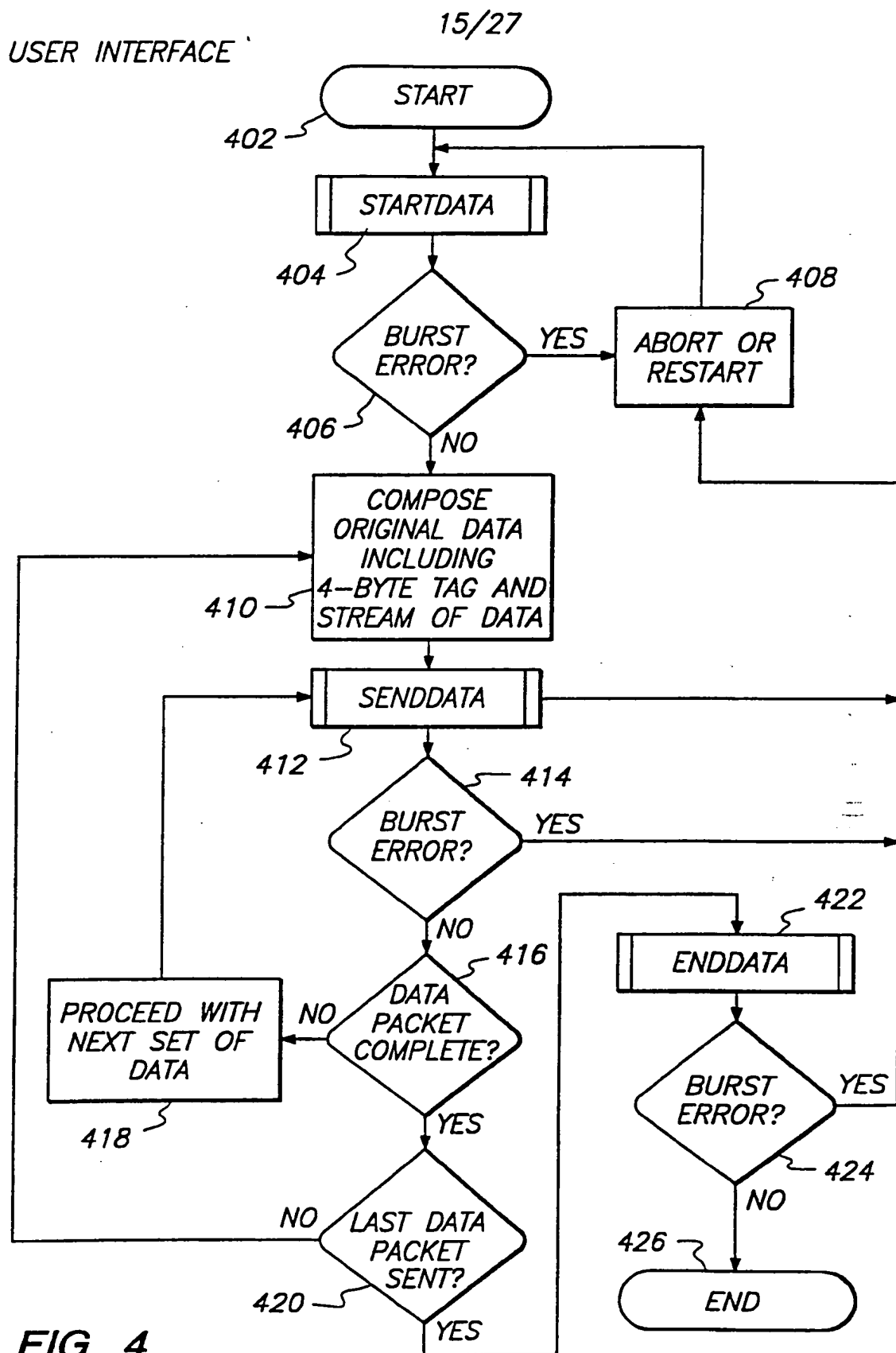


FIG. 4

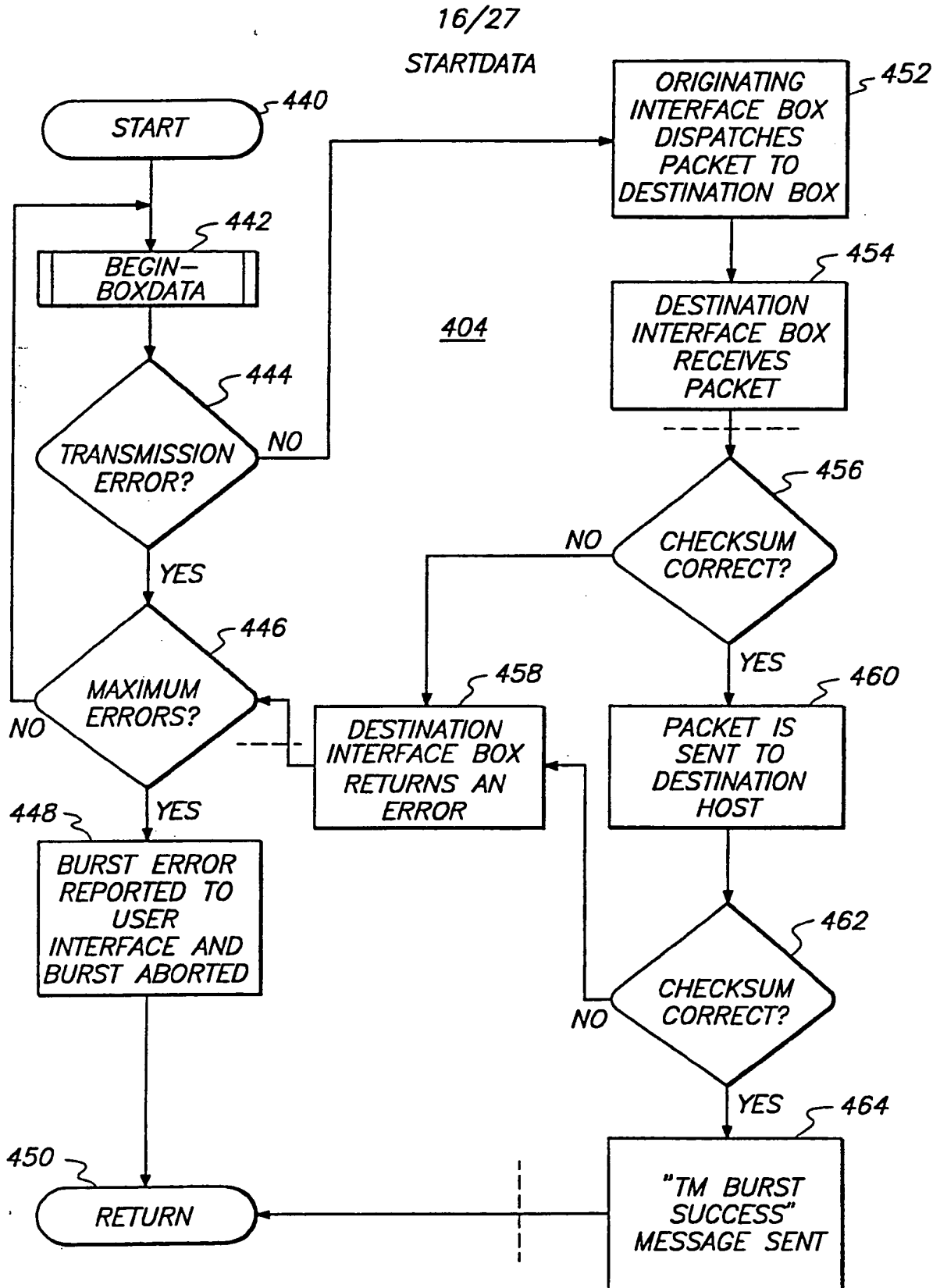


FIG. 5
SUBSTITUTE SHEET

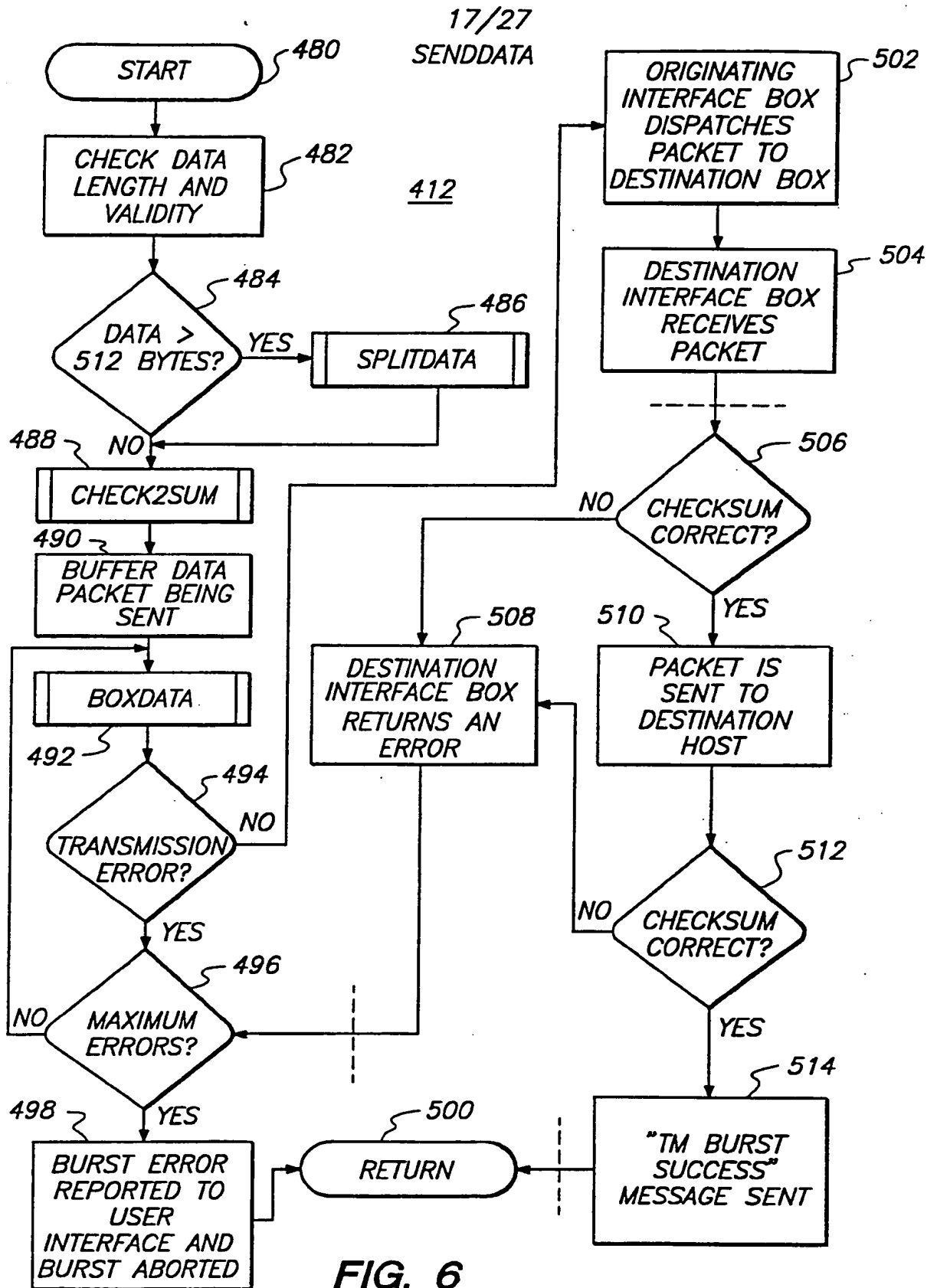


FIG. 6
SUBSTITUTE SHEET

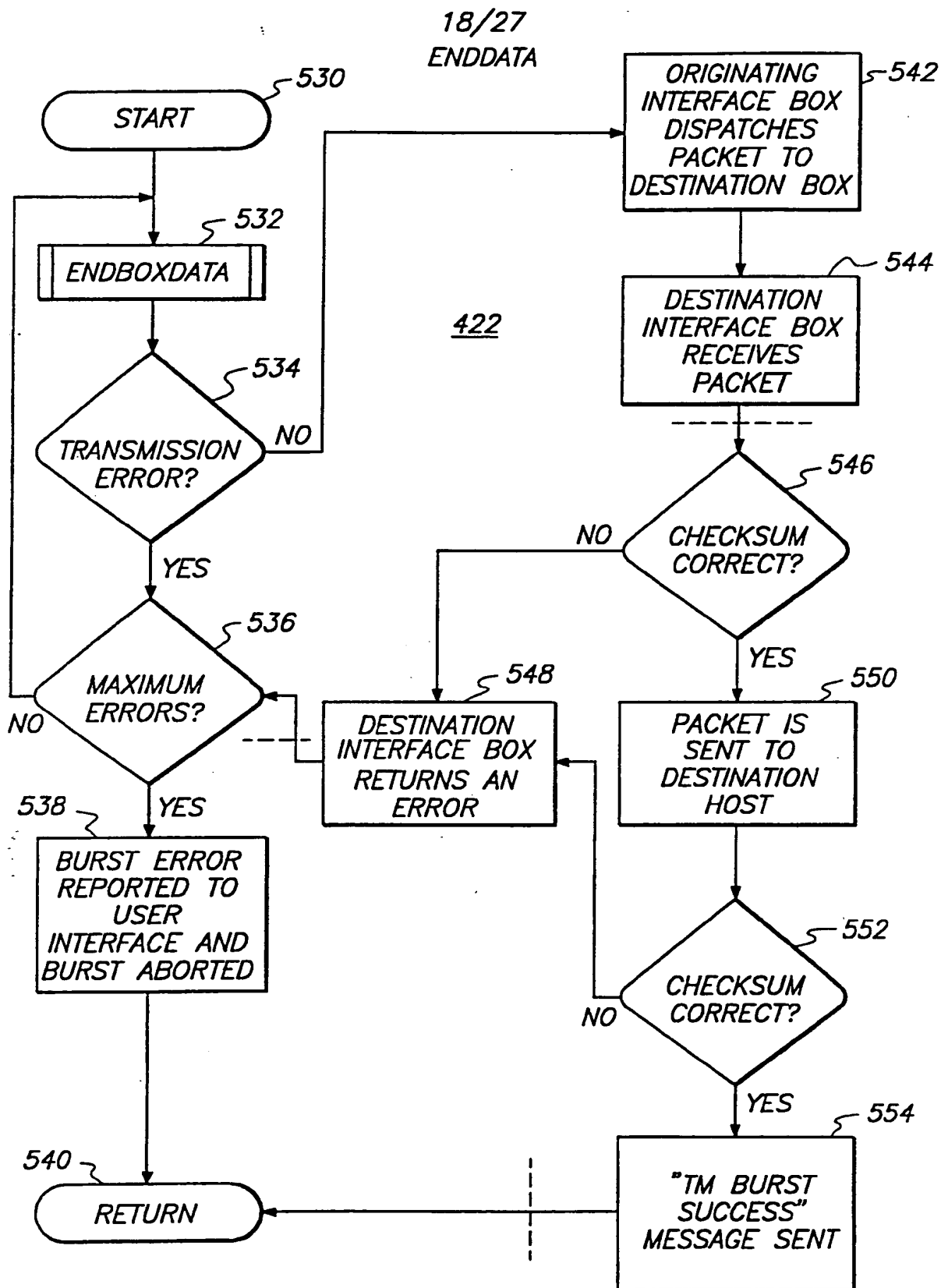


FIG. 7
SUBSTITUTE SHEET

19/27

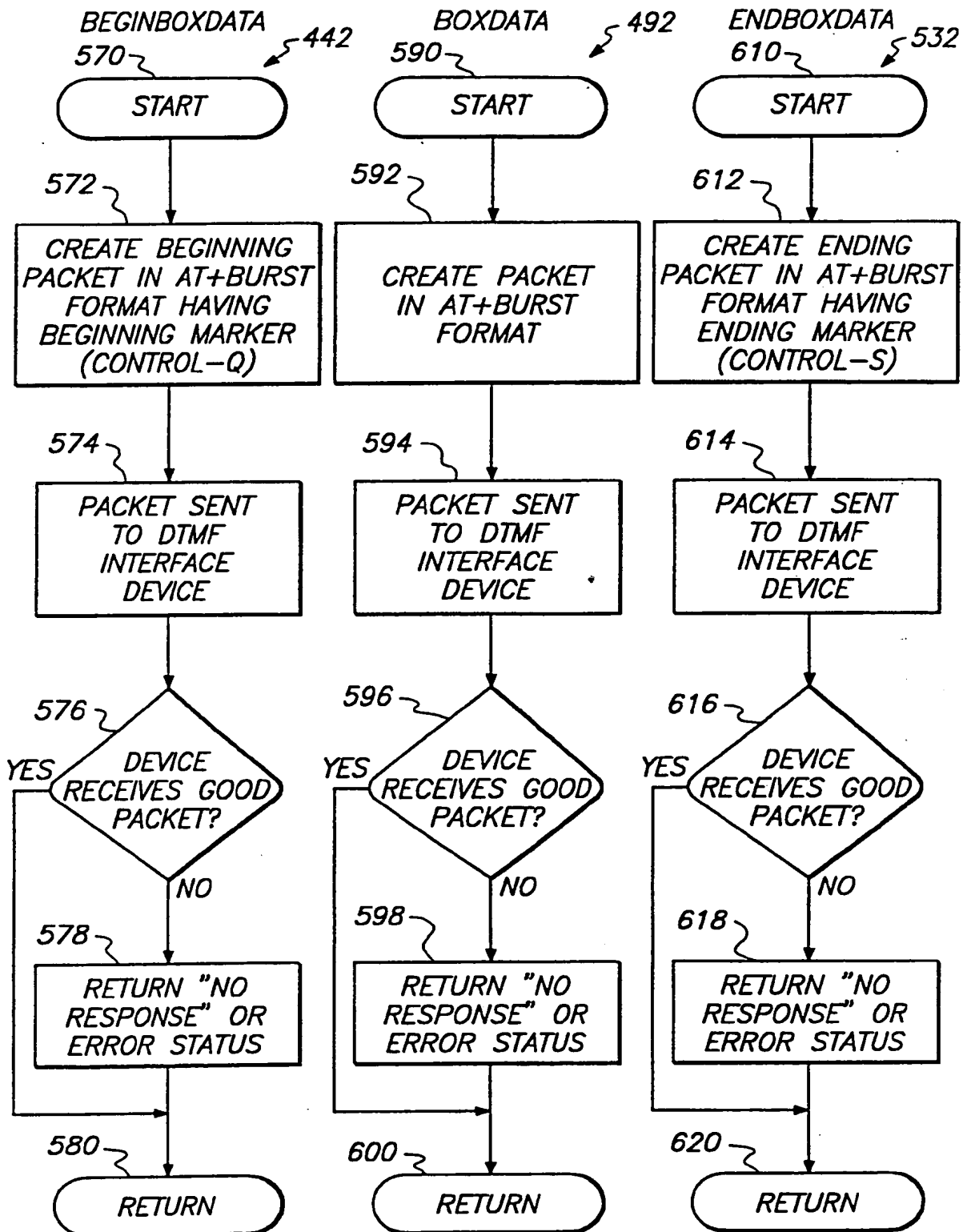


FIG. 8

FIG. 9

FIG. 10

SUBSTITUTE SHEET

20/27

TOOLS LAYER - RECEIVE

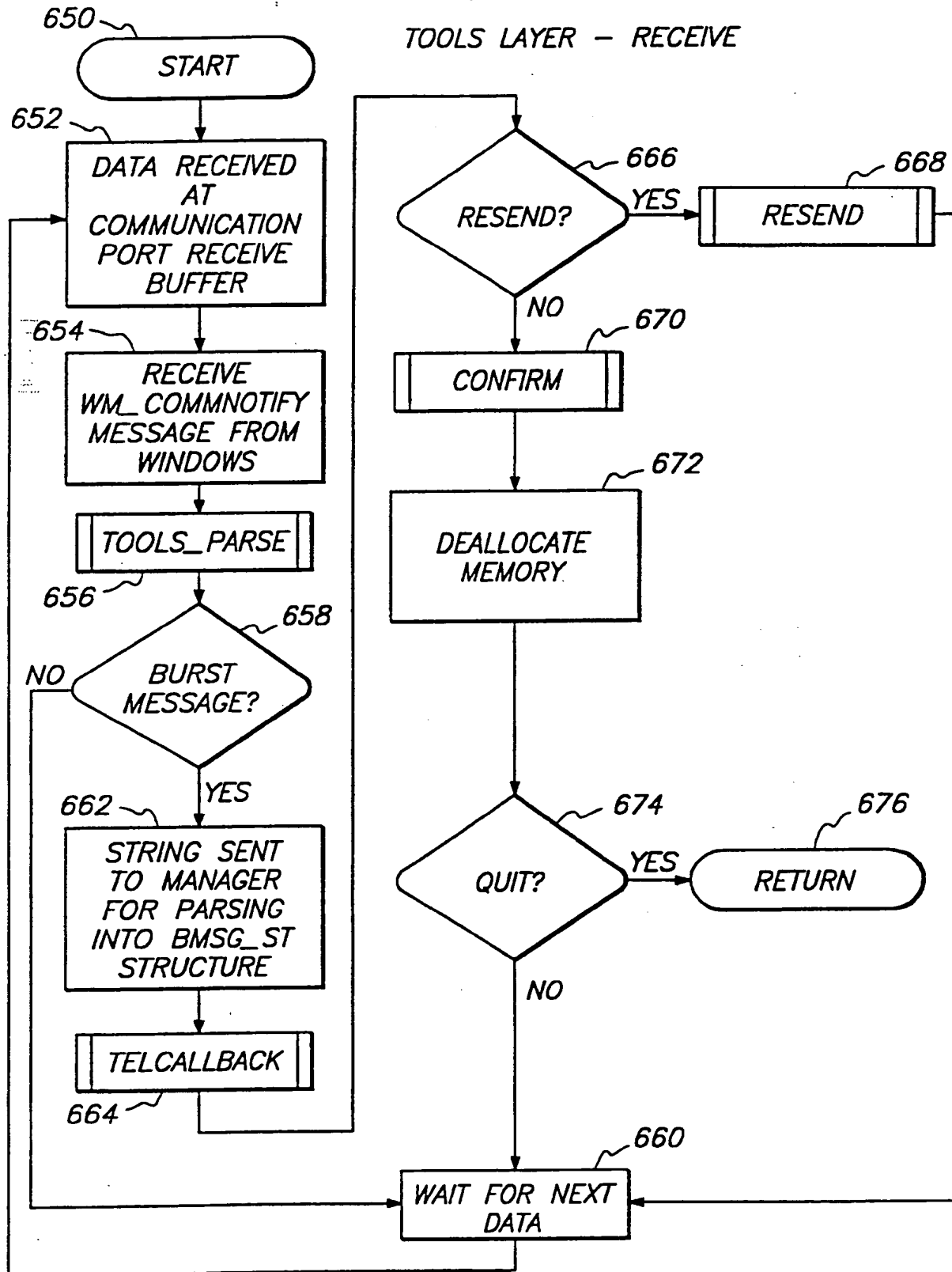
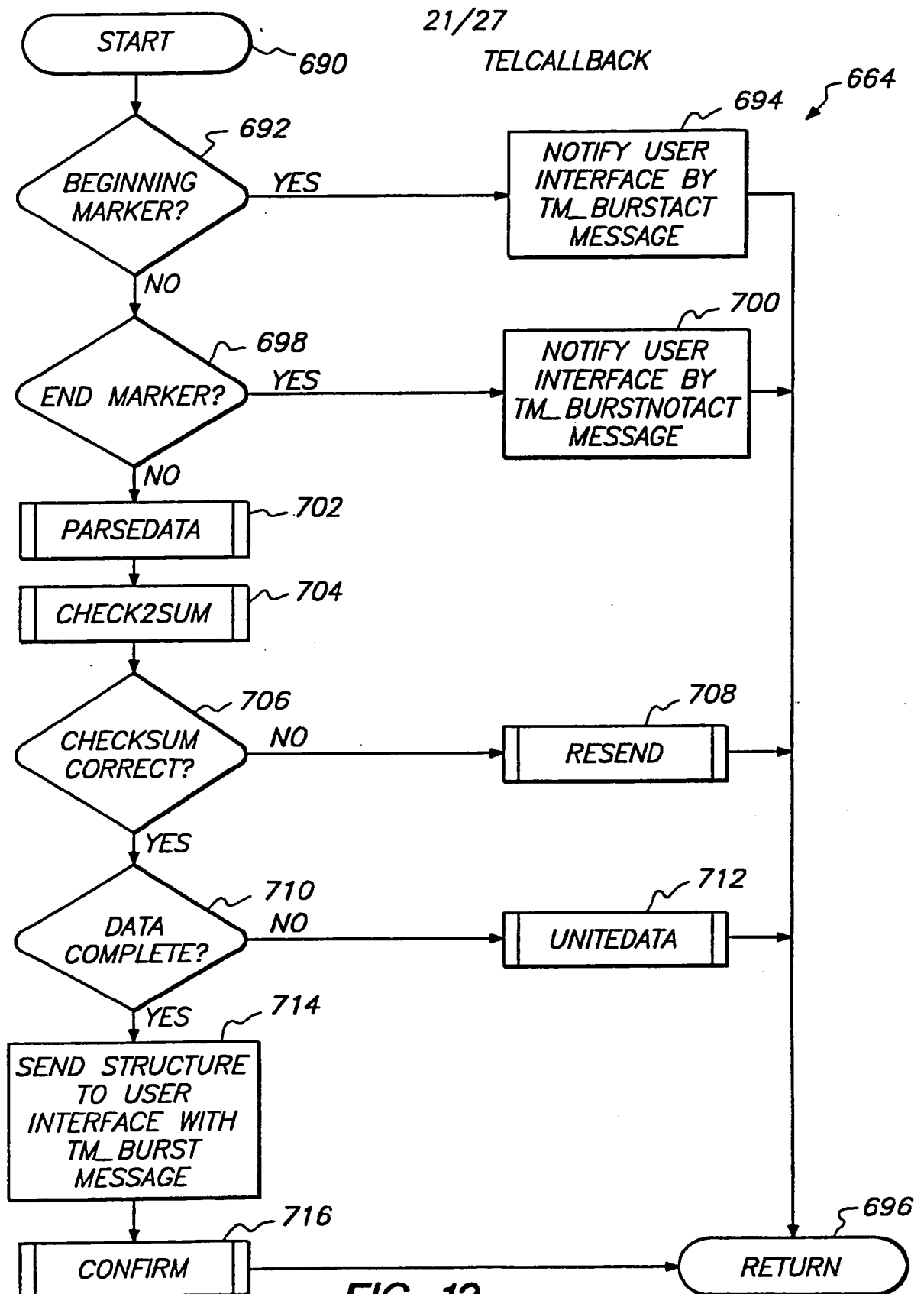


FIG. 11

SUBSTITUTE SHEET



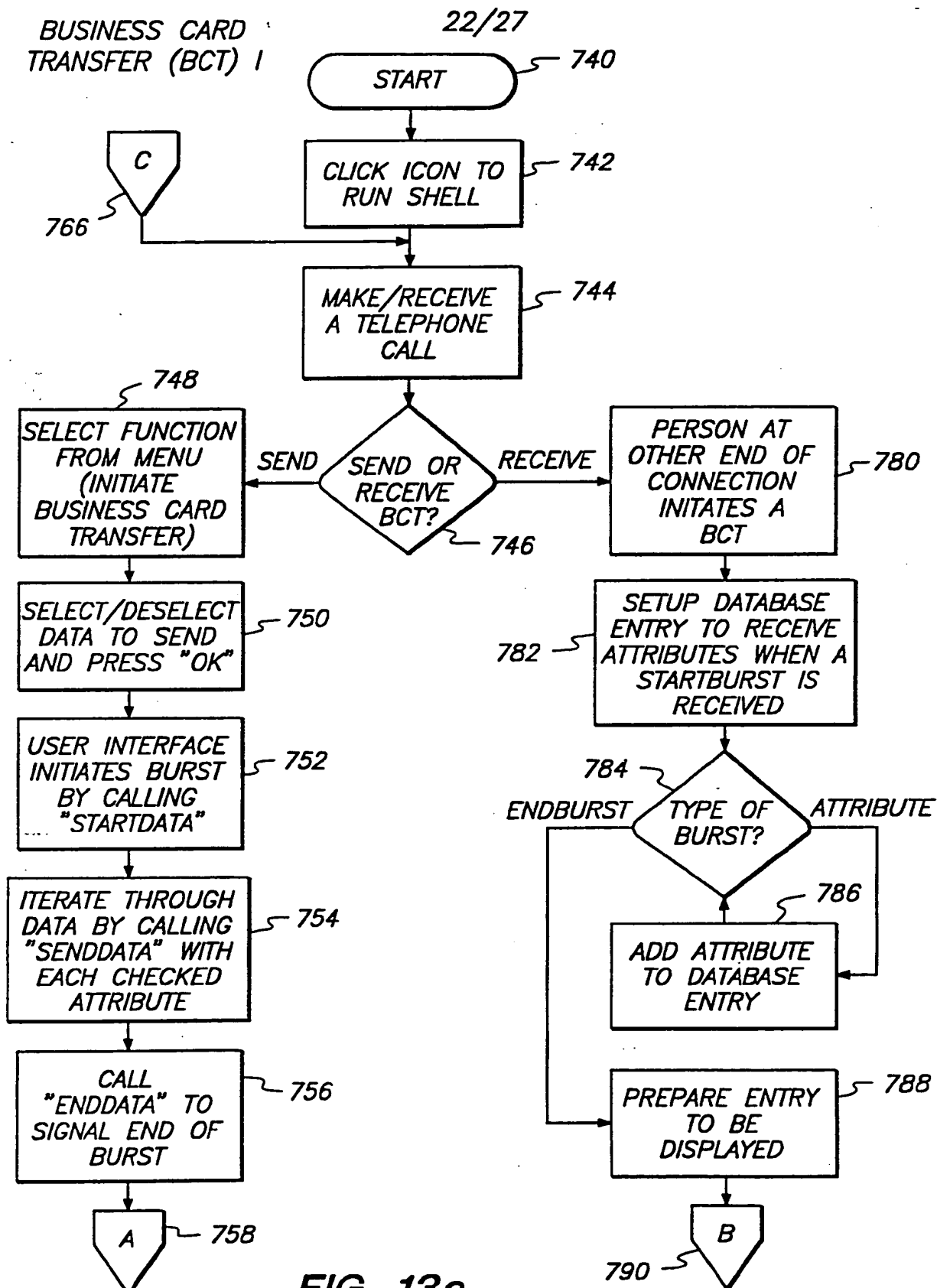


FIG. 13a
SUBSTITUTE SHEET

23/27

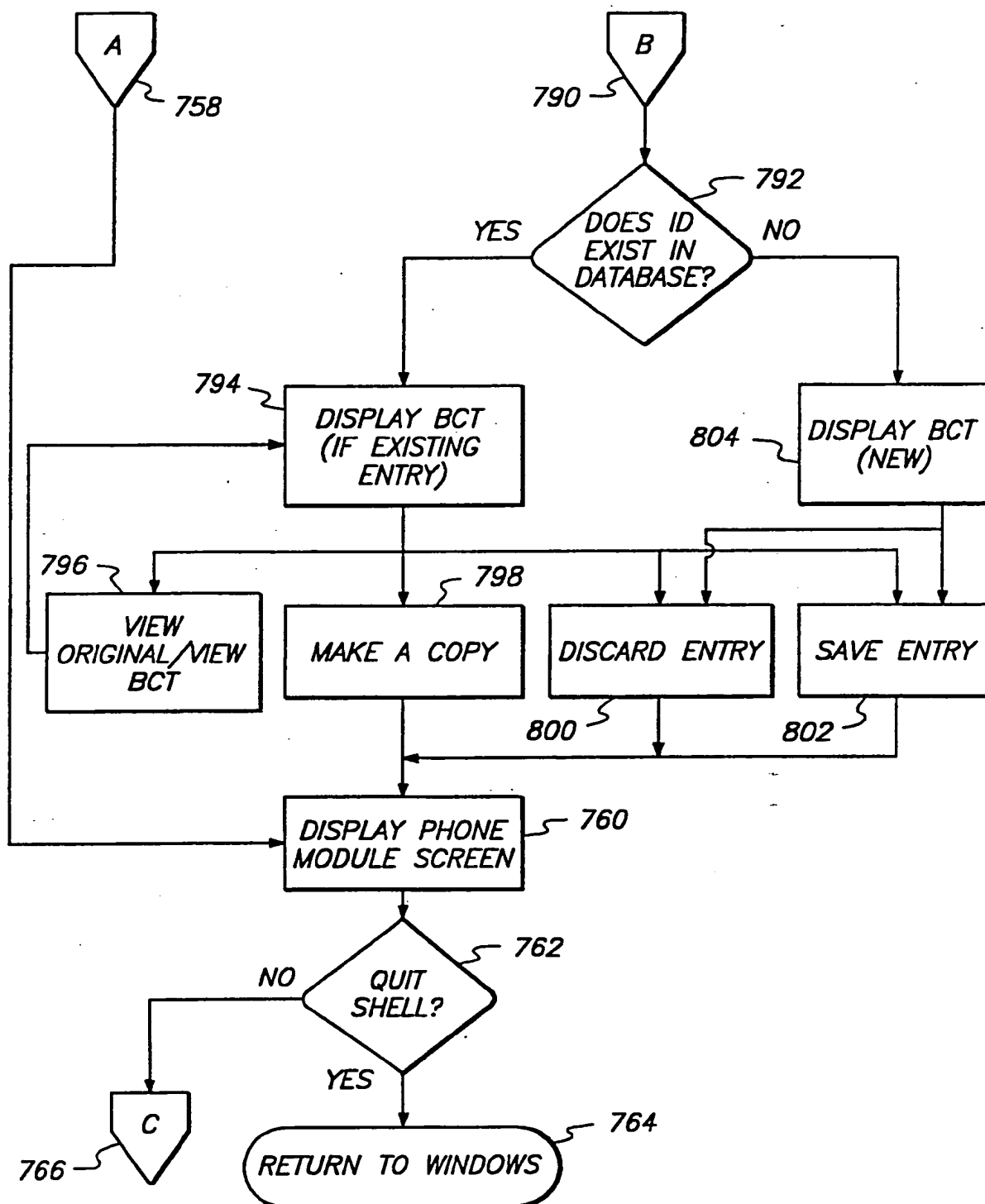
BUSINESS CARD
TRANSFER (BCT) IISUBSTITUTE SHEET **FIG. 13b**

FIG. 14

24/27

850





SubWAY v1.0			
File Edit Windows Phone			Help
 Address  Phone	Line 1	Available	00:00:00 Dial Transfer HANG UP
	Line 2	NO LINE	
<input checked="" type="checkbox"/> Contact History <input checked="" type="checkbox"/> Conference Call		Volume 5	<input type="checkbox"/> Speaker Phone Hang Up All
<div></div>		<input checked="" type="checkbox"/> Information <input checked="" type="checkbox"/> Line 1 <input checked="" type="checkbox"/> Line 2 Full Caller Information	
Add to Caller's Contact History		Delete Entry	
<div></div>			
Ready			

FIG. 15

SubWAY v1.0			
File Edit Windows Phone			Help
 Address  Phone	Kevin Hay	Connected	00:01:56 Hold Transfer HANG UP
	Line 2	NO LINE	
<input checked="" type="checkbox"/> Contact History <input checked="" type="checkbox"/> Conference Call		Volume 5	<input type="checkbox"/> Speaker Phone Hang Up All
Kevin Hay OCTuS, Inc. Engineer		<input checked="" type="checkbox"/> Information <input checked="" type="checkbox"/> Line 1 <input checked="" type="checkbox"/> Line 2 Full Caller Information	
Add to Caller's Contact History		Delete Entry	
<div></div>			
Ready			

SUBSTITUTE SHEET

FIG. 16

25/27

The screenshot shows a software window titled "SuBWAY v1.0" with a menu bar (File, Edit, Windows, Phone, Help) and a toolbar (Address, Phone). The main area displays a contact entry for "Kevin Hay" with "Line 2". A context menu is open over the contact entry, listing actions: Dial..., Answer, Hold, Retrieve, Hang Up, Hang Up All, Transfer..., Forwarding, Toggle Information, More Information..., Initiate Business Card Transfer, Contact History, and Conference Call. The status bar at the bottom indicates "Ready".

860

FIG. 17

The screenshot shows a dialog box titled "Sending Information Via octUbUrst". It contains fields for "Personal Name" (Kevin, Joe, Hay), "Business Information" (Company Name: OCTuS, Inc., Job Position: Engineer), "Company Address" (9940 Barnes Canyon Road, San Diego, California, 92121, USA), "Company Notes" (B flat.), "Office Phone Number" (1 619 452-9400 150), and "Fax Number" (1 619 452-2427). On the right side, there are buttons for "Select All", "Clear All", "Business", "Home", "Cancel", and "OK".

876 880 870 872 878 882 874

SUBSTITUTE SHEET

26/27

FIG. 18

Incoming OCTuBuRST			
First Name J		Middle Name Random	Last Name Hacker
◇ Business Information		◇ Home Information	
Company ACME		Position C.E.O.	Company Notes
Company Street Address 			
City 	State/Region 		
Zip/Postal Code 	Country 		
		C.C. (Area) 	Office Phone
		<input type="checkbox"/> Internal Ext. 	
		C.C. (Area) 	Fax Number
View Duplicate		Make a Copy	Discard Burst
			Save Burst

890 892 894 896

FIG. 19

Incoming OCTuBuRST			
First Name J		Middle Name Random	Last Name Hacker
◇ Business Information		◇ Home Information	
Company ACME		Position C.E.O.	Company Notes
Company Street Address 111 Broadway			
City San Diego	State/Region California		
Zip/Postal Code 91012	Country USA		
		C.C. (Area) 1 619	Office Phone 452-1234
		<input checked="" type="checkbox"/> Internal Ext. 	
		C.C. (Area) 	Fax Number
		Discard Burst	Save Burst

900 902

SUBSTITUTE SHEET

27/27

FIG. 20

Original Entry Existing in Database		
First Name	Middle Name	Last Name
J	Random	Hacker
◇ Business Information		◇ Home Information
Company	Position	Company Notes
ACME	C.E.O.	
Company Street Address		
111 Broadway		
City		C.C. (Area) Office Phone
San Diego		1 618 452-1234
State/Region		Ext.
California		<input checked="" type="checkbox"/> Internal <input type="checkbox"/>
Zip/Postal Code	County	C.C. (Area) Fax Number
92111	USA	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<u>V</u> iew Burst	<u>M</u> ake a Copy	<u>D</u> iscard Burst
<u>S</u> ave Burst		

SUBSTITUTE SHEET

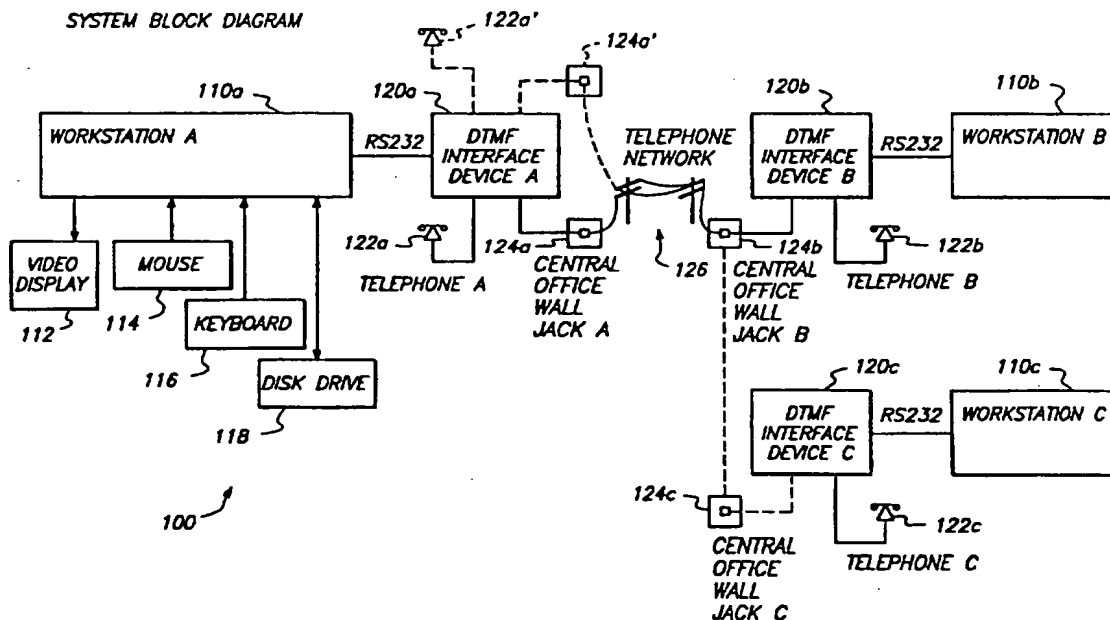
THIS PAGE BLANK (USPTO)



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁵ : H04M 11/06		A3	(11) International Publication Number: WO 94/11983
			(43) International Publication Date: 26 May 1994 (26.05.94)
(21) International Application Number: PCT/US93/11086		(74) Agents: SIMPSON, Andrew, H. et al.; Knobbe, Martens, Olson & Bear, 620 Newport Center Drive, Suite 1600, Newport Beach, CA 92660 (US).	
(22) International Filing Date: 16 November 1993 (16.11.93)			
(30) Priority data: 07/977,261 16 November 1992 (16.11.92) US		(81) Designated States: AT, AU, BB, BG, BR, BY, CA, CH, CZ, DE, DK, ES, FI, GB, HU, JP, KP, KR, KZ, LK, LU, LV, MG, MN, MW, NL, NO, NZ, PL, PT, RO, RU, SD, SE, SK, UA, UZ, VN, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).	
(71) Applicant: OCTUS, INC. [US/US]; 9940 Barnes Canyon Road, San Diego, CA 92121 (US).			
(72) Inventors: BUSHNELL, Nolan ; 3860 Woodside Road, Woodside, CA 94062 (US). ANADY, Ed ; 13862 Otis Place, Poway, CA 92064 (US). LO, Roger ; 4018 Nobel Drive #304, San Diego, CA 92122 (US). HAY, Kevin ; 11146 Caminito Inocenta, San Diego, CA 92126 (US).		Published With international search report. Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.	
		(88) Date of publication of the international search report: 29 September 1994 (29.09.94)	

(54) Title: DATA BURST SYSTEM FOR A TELEPHONE NETWORK



(57) Abstract

A system and method for transmitting data bursts across a telephone network. The present invention allows voice and data transfer during a single telephone connection. Voice communication during data transfer does not corrupt or terminate the transfer. The present invention includes an interface device for encoding and decoding DTMF tones into data. Caller information can be selectively transferred by a telephone user to the called person.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	GB	United Kingdom	MR	Mauritania
AU	Australia	GE	Georgia	MW	Malawi
BB	Barbados	GN	Guinea	NE	Niger
BE	Belgium	GR	Greece	NL	Netherlands
BF	Burkina Faso	HU	Hungary	NO	Norway
BG	Bulgaria	IE	Ireland	NZ	New Zealand
BJ	Benin	IT	Italy	PL	Poland
BR	Brazil	JP	Japan	PT	Portugal
BY	Belarus	KE	Kenya	RO	Romania
CA	Canada	KG	Kyrgyzstan	RU	Russian Federation
CF	Central African Republic	KP	Democratic People's Republic of Korea	SD	Sudan
CG	Congo	KR	Republic of Korea	SE	Sweden
CH	Switzerland	KZ	Kazakhstan	SI	Slovenia
CI	Côte d'Ivoire	LI	Liechtenstein	SK	Slovakia
CM	Cameroon	LK	Sri Lanka	SN	Senegal
CN	China	LU	Luxembourg	TD	Chad
CS	Czechoslovakia	LV	Latvia	TC	Togo
CZ	Czech Republic	MC	Monaco	TJ	Tajikistan
DE	Germany	MD	Republic of Moldova	TT	Trinidad and Tobago
DK	Denmark	MG	Madagascar	UA	Ukraine
ES	Spain	ML	Mali	US	United States of America
FI	Finland	MN	Mongolia	UZ	Uzbekistan
FR	France			VN	Viet Nam
GA	Gabon				

INTERNATIONAL SEARCH REPORT

Inter national Application No
PCT/US 93/11086

A. CLASSIFICATION OF SUBJECT MATTER
IPC 5 H04M11/06

According to International Patent Classification (IPC) or to both national classification and IPC.

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 5 H04M

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X Y	US,A,4 860 342 (DANNER) 22 August 1989 see column 2, line 56 - column 3, line 39 see column 4, line 17 - line 38 see column 4, line 52 - line 57 see column 5, line 16 - line 24 see column 5, line 66 - column 6, line 2 see figures 1,2,4 ---	1-6,8-17 7
Y	US,A,4 864 601 (BERRY) 5 September 1989 see column 4, line 12 - line 14 ---	7
A	EP,A,0 374 828 (ALCATEL BUSINESS SYSTEMS) 27 June 1990 see abstract --- -/--	1,10,14

☒ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

* Special categories of cited documents :

- *A* document defining the general state of the art which is not considered to be of particular relevance
- *E* earlier document but published on or after the international filing date
- *L* document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
- *O* document referring to an oral disclosure, use, exhibition or other means
- *P* document published prior to the international filing date but later than the priority date claimed

- *T* later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
- *X* document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
- *Y* document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.
- *&* document member of the same patent family

Date of the actual completion of the international search

5 August 1994

Date of mailing of the international search report

18.08.94

Name and mailing address of the ISA

European Patent Office, P.B. 5818 Patendaan 2
NL - 2280 HV Rijswijk
Tel. (- 31-70) 340-2040, Tx. 31 651 epo nl,
Fax (- 31-70) 340-3016

Authorized officer

Goossens, A

INTERNATIONAL SEARCH REPORT

Int. Patent Application No.

PCT/US 93/11086

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US,A,5 097 528 (GURSAHANEY ET AL.) 17 March 1992 see abstract see figures 11-14,17 -----	1,10,14

INTERNATIONAL SEARCH REPORT

information on patent family members

International Application No

PCT/US 93/11086

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US-A-4860342	22-08-89	NONE	
US-A-4864601	05-09-89	JP-A- 2016858	19-01-90
EP-A-0374828	27-06-90	FR-A- 2641093	29-06-90
		DE-D- 68914961	01-06-94
		ES-T- 2052880	16-07-94
		JP-A- 2224563	06-09-90
		US-A- 5065425	12-11-91
US-A-5097528	17-03-92	EP-A- 0501189	02-09-92
		JP-A- 4353957	08-12-92

THIS PAGE BLANK (USPTO)